Title: **A Study on Virtual Reality and Developing the Experience in a Gaming Simulation**

Name: LIONEL JAYARAJ

A Study on Virtual Reality and Developing the Experience in a Gaming Simulation

LIONEL JAYARAJ

MSc

IRAC

2016

UNIVERSITY OF BEDFORDSHIRE

A Study on Virtual Reality and developing the experience in a Gaming simulation.

by

LIONEL JAYARAJ

A thesis submitted to the University of Bedfordshire in partial fulfilment of the requirements for the degree of Masters by Research

March 2016

Abstract

Virtual Reality (VR) is an experience where a person is provided with the freedom of viewing and moving in a virtual world [1]. The experience is not constrained to a limited control. Here, it was triggered interactively according to the user's physical movement [1] [2]. So the user feels as if they are seeing the real world; also, 3D technologies allow the viewer to experience the volume of the object and its prospection in the virtual world [1]. The human brain generates the depth when each eye receives the images in its point of view.

For learning for and developing the project using the university's facilities, some of the core parts of the research have been accomplished, such as designing the VR motion controller and VR HMD (Head Mount Display), using an open source microcontroller. The VR HMD with the VR controller gives an immersive feel and a complete VR system [2].

The motive was to demonstrate a working model to create a VR experience on a mobile platform. Particularly, the VR system uses a micro electro-mechanical system to track motion without a tracking camera. The VR experience has also been developed in a gaming simulation.

To produce this, Maya, Unity, Motion Analysis System, MotionBuilder, Arduino and programming have been used. The lessons and codes taken or improvised from [33] [44] [25] and [45] have been studied and implemented.

Keywords: 3D Modelling, 3D Animation, Games Development, Interactive Technologies, Motion Capture, Real-time Animation, Motion Sensors and Filters.

Declaration

I declare that this thesis is my own unaided work. It is being submitted for the degree of MSc at the University of Bedfordshire.

It has not been submitted before for any degree or examination in any other University.

Name of candidate: Lionel Jayaraj Signature:

Date:

Contents

Table of Figures

List of Abbreviations

2D	Two Dimension
3D	Three Dimension
3DOF	3 Degrees Of Freedom
6DOF	6 Degrees Of Freedom
9DOF	9 Degrees Of Freedom
AHRS	Altitude Heading Reference System Algorithm
AMR	Anisotropic Magnetoresistive
AR	Augmented Reality
CG	Computer Graphics
COM	Communication
DMP	Digital Motion Processing
GND	Ground
GPIO	General-Purpose Input/Output
GPS	Global Positioning System
GUI	Graphical User Interface
HMD	Head Mount Display
IC	Integrated Circuit
IK	Inverse Kinematics
IMU	Inertial Motion Unit
IR	Infra-Red
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LION	Lithium-Ion
MARG	Magnitude Angular Rate and Gravity device
MEMS	Micro Electro Mechanical System
MHL	Mobile High-Definition Link
MITO	Monsters in the Orchestra
MOCAP	Motion Capture
MPU	Motion Processing Unit
OTG	On The Go
PCB	Printed Circuit Board
Pix	Pixel
PLA	Polylactic Acid
SDA	Standard Data
SDC	Standard Clock
SME	Small and Medium Enterprise
Tex	Texture
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
UV	U curve and V curve
VCC	Power Supply
VR	Virtual Reality
VRHMD	Virtual Reality Head Mount Display

Acknowledgments

I would like to thank and acknowledge my supervisor Dr. Jim Wood, who is the best in the School in hardware interfacing, for guiding and encouraging me on my project and my second supervisor Mr. Rob Manton, who specializes in the School with the games development. Apart from them, I have had tremendous support from Mr. David Jazani, Mr. Mohamed Alom, and Mr. Mark Hooper, who have inspired me with their commitment to bring out my ideas working. My colleagues Grigor Trodov, Joshua Anderson, Zachary Cohen and Yi Zu (Joey) are my greater support throughout the research. I would like to acknowledge Dr. Peter Norrington for helping me to edit my writing academically. Special thanks goes to Grigor Trodov, Zachary Cohen, and Joshua Anderson, who have gone the mile extra in teaching me and storming new ideas, on top of developing their own MSc research projects. For the research, I have gathered most of the knowledge and ideas through books, journals, documents, tutorials and also through the interaction in the society of researchers, facilities, working space and internet provided by the University of Bedfordshire. I would like to mention my parents for supporting my dreams and endeavours.

Introduction

Virtual reality (VR) experience is predominantly used on workstations and for console gaming. Mobile VR should be the next generation immersive gaming [2]. The current commercial models are called 'Gear VR' which is adapted from Google Cardboard technology. These use the sensors built into the mobile to track the head, use the phone screen as a display and a joystick to control the game.

The affordability of commercial VR is expensive. Meanwhile, mobile phone technology has started to come with a powerful processor that supports multi-threading [1].

The proposed solution gives a novel approach to the current system, to provide the user with a system for a complete VR experience on the mobile phone.

Aim and Objectives

Aim

To design an approach to VR game control suitable for a mobile platform.

Objectives

1. To assess existing VR game controls
2. To design a VR game control suitable for a mobile platform
3. To test the mobile VR game control
4. To evaluate the proposed design against existing designs

Problem to resolve

There is no point in playing a VR game on a mobile using a regular game controller. So the intention is to implement a VR controller for mobile games that tracks the motion and triggers instant animation and controls [3]. The solution should be affordable to the general market.

Context

The researcher's journey started with an idea of improving the controls used for virtual reality gaming. The reason was that the concept of a virtual reality should work in both these ways:

- bringing the virtual world into reality and also
- bringing reality into the virtual world.

Literature review and research into motion capture technology has been conducted and implemented in tracking in a real-time game. The technology adopted here could be the future [4] of the animation, gaming and robotics industry because it provides a real-time non-head-tracking motion capture at an affordable price using the Micro Electro Mechanical System (MEMS) [5].

The research includes the traditional approach to motion capture using optical technology [6] and the software pipeline between motion capture and game production. The researcher taught students and utilized the opportunity for evaluation. The project provides a MEMS Motion controller, HMD for Windows/Android and adopted a complimentary filter to estimate the orientation. Finally, the users' experience was tested and data collected.

Contribution of the work

- The VR was designed to plug a HMD (Head Mounted Display) to a mobile phone and is played using the phone's motion. The HMD is free from any camera that tracks the head.
- The design uses the phone as the game controller which is superior to the regular joystick because the phone can be used as a motion controller which should be the most compatible controller for a VR game and also give a live capture of the body movement.
- A specialized motion controller with joystick is provided to the project which is meant to be handy to avoid the risk of handling the phone and also with the intention of using it as a left hand controller to track the other hand.
- The design has been tested with views collected from students at the University of Bedfordshire.

Literature Review

Use of virtual reality for games

Roquilly studied the use of virtual reality and provided components to experience virtual reality in a game. He further states that his study could inspire future researchers and innovators. Hence, there are areas to investigate and improve virtual reality for computer games [4].

Motion controllers for games

Adrian [3] discussed the necessity of innovation in gaming in his article. He states that modern games will employ motion controllers and technologies similar to Microsoft Kinect with tracking devices to input the game and a gyroscope that can track the movement of the head in a virtual world. Hence, this researcher believes motion controllers in virtual reality simulations will be the stepping stone for next generation gaming. Android phones can generate virtual reality similar to Oculus Rift using Gear VR.

Interaction with future games

The SIGGRAPH ’14 “Special Interest Group on Computer Graphics and Interactive Techniques Conference” [7] discussed various technologies in relation to futuristic interaction in computer graphics. This conference inspires many ideas and projects. The topics that are of interest from [7] are:

1. A solution for a see-through monitor with the interaction.
2. A glassless television for 3D movies.
3. A mechanical tracking system with an HMD for a Birdy attempting to fly.
4. Buru-Navi3 technology that measures a pull between two sensors.
5. Cyberith Virtualizer – A device for virtual reality to give friction-free locomotion.
6. A touch or a gesture response for Kinect
7. A drone which is controlled by gesture and proximity.
8. The HaptoMirage System for an interactive 3D display.
9. A motion tracking system based on magnetism.
10. Lights as a source for the communication (LIFI).

11. Monsters in the Orchestra (MITO) – a VR concert.
12. A see-through AR Google with a pin light LED source.
13. A modular rig that interfaces the character.

Speech: the next control for VR games

Speech should be the next level of interaction in a VR game. A player who accomplishes a task using a speech command will have an additional immersive experience in a VR game rather than a player using keys. The technology reduces the usage of a key to controlling the VR [4].

Since a player cannot see the controls in the virtual world, this researcher believes speech control can be good for VR gaming [7].

Use of speech commands in computing

Freitas and Kouroupetroglou studied the use of speech recognition systems for those who are blind and discussed the interface in connecting them with computers to access the World Wide Web [7].

Recently, Calandruccio and Haibo, in their paper [8], talked about the need for improvement in speech recognition to provide multilingual ability and to avoid mismatch between the target. Hence, there are unusual methods already available to take speech as a command in target computing.

Voice controls are available in Unity to design a game with simple voice commands like “start, open, close, stop” to control animation. The script in C# is available on the internet to take voice as command.

There are technologies like the vocal joystick to provoke animation with speech commands. Hence, specialised VR-based speech interaction can be achievable as observed in the article by Kurniawan and Sporka in 2008 [9].

Some Unity plug-ins related with voice to digitalisation are:

- Speecher: https://www.assetstore.unity3d.com/en/#!/content/3021
- Speech Recognition for Android: https://www.assetstore.unity3d.com/en/#!/content/13882 and
- Intel® Perceptual Computing SDK Plug-in: https://www.assetstore.unity3d.com/en/#!/content/16241

Use of sensors for motion tracking

Motion capture is also achieved using MEMS (Micro Electro Mechanical System) Technology [5]. The technology is one of the non-optical methods where the usage of the camera is compensated. The MEMS technology provides various sensors for different purposes. Madgwick [7] describes two devices that can track movement – Inertial Motion Unit (IMU) and Magnitude Angular Rate and Gravity (MARG) devices. These devices are sensor based.

In his report, he also speaks about the method's adaptation from the Kalman Filter [8], which is the testament for various innovations in the field of robotics and animation technology. Madgwick's approach simplifies the complexity in the Kalman Filtering.

He also evaluated his method [7] by doing experiments comparing his devices with the reliable optical tracking method and achieved a favourable result.

Artefacts

Introduction

As per the definition given by Google, “Artefacts” are referred to as “an object made by a human being, typically one of cultural or historical interest.” This chapter discusses the artefacts created for the virtual reality project.

The intention is to use this technology in a sporting game simulation to be played by hand swing. The idea was to develop a VR game for the “Indian Premier League” which is a franchise-based cricketing tournament that is celebrated across India in the summer vacation. Development of a game helps with learning skills to understand the game creation pipeline and the hardware this interacts with.

The proposed VR system is intended to play the game using mobile technology. The artefacts for this project can be categorised into:

- 3D artefacts
- Animation artefacts

3D Artefacts

Cricket in a virtual reality game

The particular game initially chosen for development is limited-over cricket, which would be a fascinating experience because cricket is celebrated across India and every Indian is crazy for the game.

Also, another reason for choosing a VR-based sporting game is consciousness of health and safety. There are two factors to be concerned about:

1. Motion sickness
2. Health and Safety

Motion sickness

Motion sickness is the mental state in which the human brain struggles to co-operate with the interpolating visuals from the VR goggle [2]. When the researcher tried the Oculus Rift DK2, initially, he felt dizzy and the symptoms of nausea for some games which have an involuntary movement towards the front.

The movements are according to the angle of the heading. The game should not be suggested to children. The research has never enjoyed testing those games, even feeling an unusually pumping heart and choking breath.

Even for the mature person playing those games, the researcher would suggest developers to make a frame on the gaming screen that helps the user to focus [3]. The delay in the slerp (interpolating curve paths from A to B) also influences sickness. The delay could be because of factors like processing power, insufficient graphics memory, etc. The inappropriate usage of motion blur and low frame rate can also cause motion sickness, in this researcher's opinion.

Health and safety

It is not advisable to hold a phone to the head for hours even though the phone has a display and also comes with the gyroscope [5], which includes all the sensors together that track the movement of the phone. This leads to suggesting a novel approach to control a mobile game with the motion of the phone.

The two primary reasons for suggesting health and safety concerns about mobile phones as not safe for VR are:

- The lithium-ion battery can be hazardous. It has a tendency to explode when the charge is drained or overcharged or used while charging. These are reported as a rare occurrence. Designing a device without a battery that utilizes the power from a mobile phone which should be kept away from the head should be the best solution.
- Radiation from radio and electromagnetic signals in a cell phone can affect brain neurons when regularly kept in contact. Video games are addictive, and one can spend hours and hours on playing games. This reason led the researcher to an idea to design a different VR goggle which is meant to connect to the mobile phone via cables that receive the power and signals from the phone. So the phone is not kept in contact with the head.

Artefacts for the game

Artefacts comprised more than three-quarters of the researcher's course work spent in developing the objects for the computer game. The researcher believes that artefacts are more important in the project because it is about creating the virtual world and wanted to show how a virtual world was achieved from scratch.

Like in the real world, the virtual world should have the 3D prospection. The 3D models can be categorised as environments, characters and props. Apart from the 3D models, there are key areas to concentrate on, like materials, lights and shadows, ambiance and the mood of the scene.

As per current 3D technology, it is possible to model a virtual world resembling the same as the real world [13]. A team of artists and months of hard work will make it possible. As a result, a viewer can believe that fantastic computer graphics works from Hollywood are real.

In this case, the researcher is the lone worker for the project, and has achieved what he is capable of delivering in the given span of time. Of nine months of working on the artefacts, the works are categorised into three quarters.

- 1st Quarter: **Character and Environments**
- 2nd Quarter: **Motion Capture**
- 3rd Quarter: **Pipelining a Game Development**

Characters and environments

As per the gaming standards [14], the model should be made more wisely because the details and the sculpture should not be made of polygons (the cells by which 3D models are made of).

Here, the details rely more on the texture maps and the materials used for a low polygonal model [15]. Hence, all the models are made in low polygon model for an efficient animated gaming experience.

This game is meant to run on an Android platform which can render 500K polygons per scene. There was a necessity to create a view matching the Android standards, which are not exceeding more than 500K polygons in each frame. The object in the front of the camera should have denser polygons than the object beyond [16]. Most commonly, the objects in the background are given less treatment than the objects in the foreground. Also, the priority is given based on the area of concentration.

Usually, the character is concentrated on more. Especially the face should have more details for a sporting game because it differs from player to player. Here, the same body is used for all the characters. The texture has been changed to make the job easier.

For a test, an initial model was created for the game. It looks like an amateur finish, but the traditional style of the [17] rig was tested on it. This has foot control, reverse foot chain, inverse kinematics, set driven key and a pelvis control [14] which derives the whole body as the user moves their foot control to either front or back.

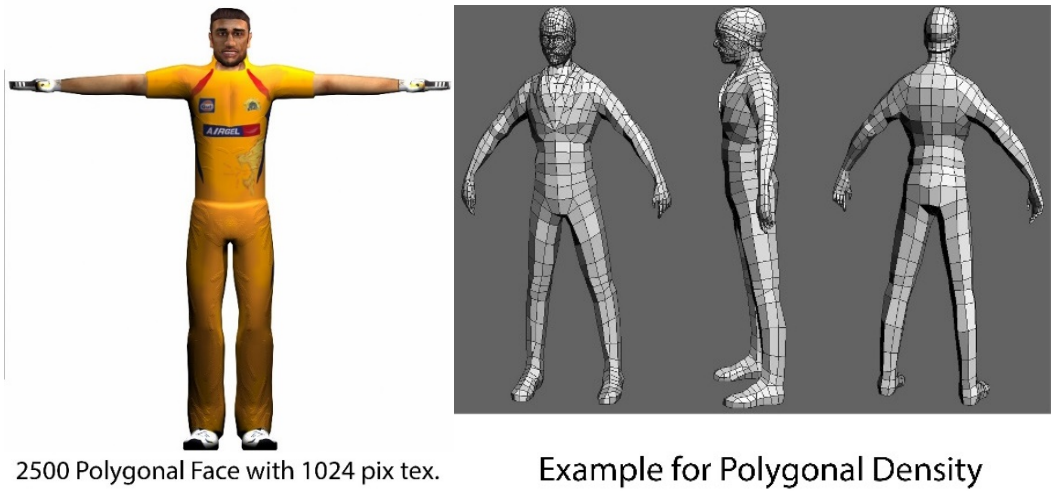
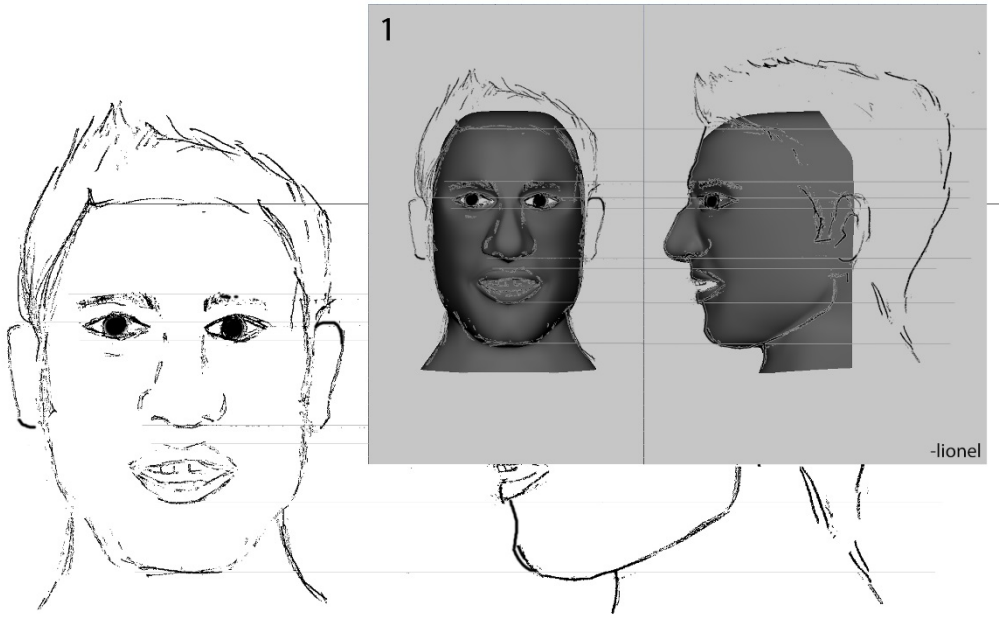
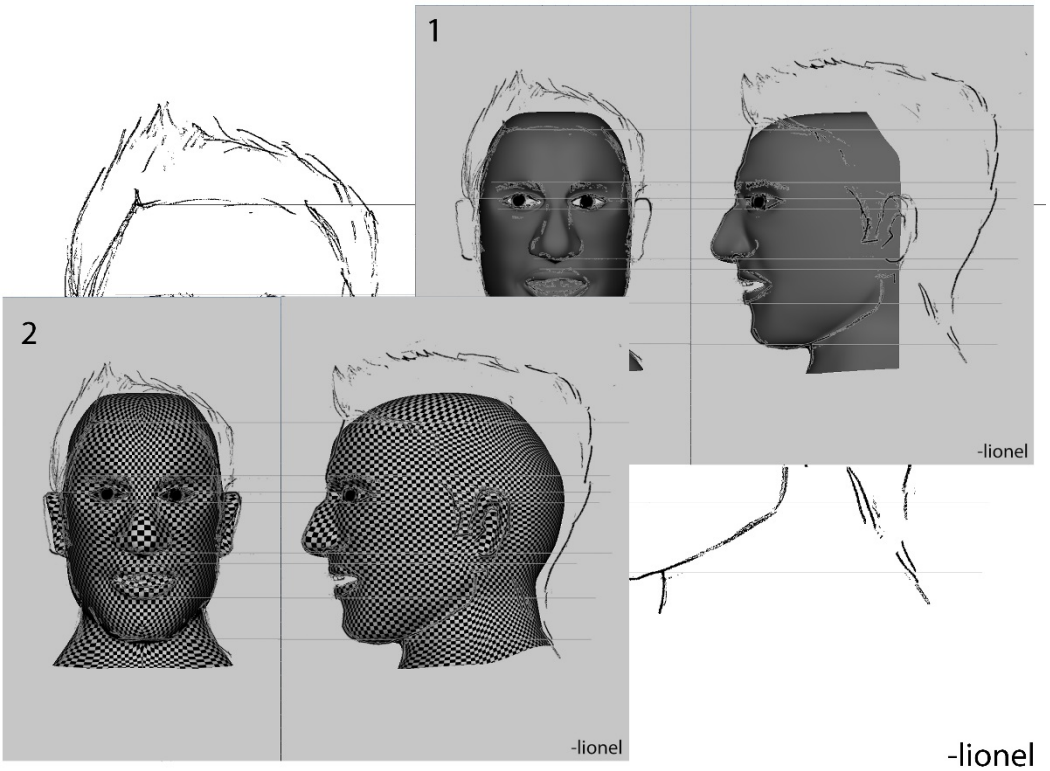


Figure 1: Modelling restriction For Games

The progression is in chronological order, and the picture sequence below explains this.



-lionel



-lionel

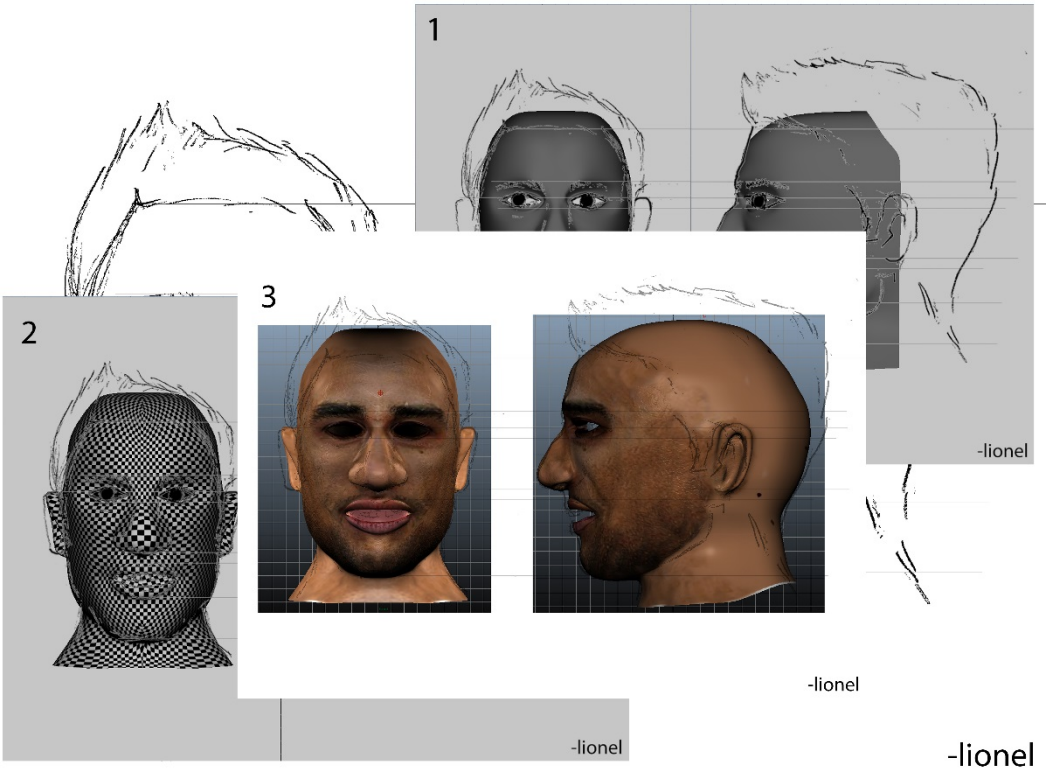


Figure 2: A Modelling Progression

The drawing was based on a real character [18] who plays cricket for India. A photo of him has been used to texture his face.

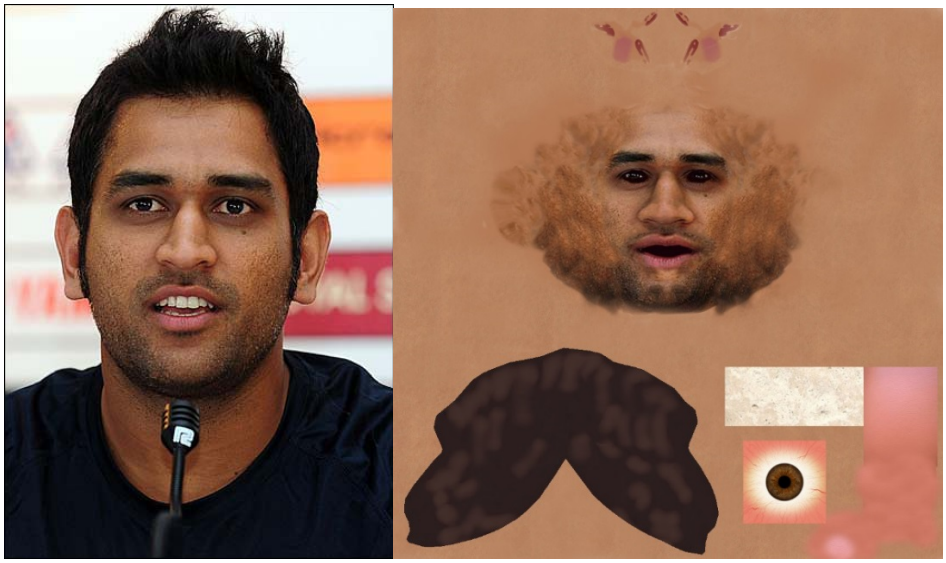


Figure 3: Texturing for the Model

This test satisfied the appropriate usage of polygons and textures according to the gaming standards [20]. With this attempt, the pipeline was successfully

tested, but the finish is an amateur one. This received much negative feedback, so this work was set aside and the researcher pushed on to the next level researching human anatomy [19].

The next step was to create a model which meets the logic of anatomy [18] [13] [19]:

- Human height is eight times of own head.
- A gap of an eye's width should be between both eyes.
- The candid and side portrait are the best angles to judge a form.
- Nose and ear end fall in the same line.
- Similarly, the nose and ears beginnings fall on the same line.
- Hands end halfway between the knees and hip.
- Forearm has same distance as back arm.
- Similarly, thighs have the same distance as the legs.
- Reference diagram should resemble the character at least 85%.
- Reference diagram should have details falling on the same line.

Considering all the points above, the researcher studied and referenced each and every character face and sketched them initially to perfection. For the body, a human dummy was taken and adjusted for the character's limbs and body accordingly. The study has corrected the researcher's deficiency in anatomy.

Five key members per team were selected and models started for them: they are the main characters in the game. They will be in the foreground within the radius of the camera. The players at the boundary are duplicates of the eight characters. After sketching them, the same pipeline [21] was followed to complete the model.

3D modelling pipeline

A plane on the X-axis and Z-axis was mounted. The front view of the reference sketch was loaded to the plane facing the Z-axis, and the side view of the reference drawing loaded to the plane facing the X-axis. It is advisable to change the image into a .png format that has only the sketch visible, and the white areas

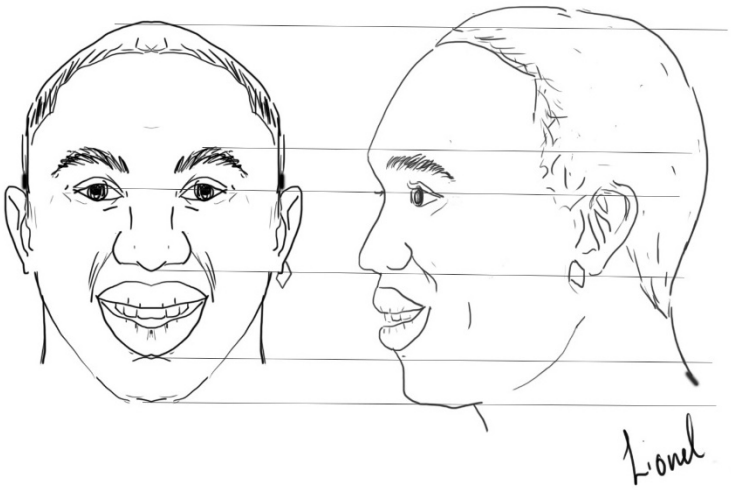
were made transparent which are locked in the layers. The method gives more accuracy for a 3D artist and 100% assurance that the models are as per the guidelines [16].

The pipeline was shaping one half of the face initially. On modelling one-half of the face [15], the other half is achieved by mirroring; here usually the left half of the face is formed and the right side mirrored. The next step is to map the U curve and V curve (UVs) [14]. Once the UVs are mapped, the UVs are tested by applying textures like checker pattern to check whether the UVs are evenly distributed.

After mapping the UVs, the texturing is done either in Photoshop or Mudbox [16]. Mudbox and Photoshop together here provide a fabulous result. Using MudBox, the normal, displacement and ambient occlusion maps were generated. Taking these maps into the Photoshop, more details were added to the generated textures. The final surface is saved as a .jpeg image.

After modelling and rigging all the characters who are playing the game, the environment [21] was modelled. For this, two stadiums were taken as a reference, namely M.A. Chidambaram Stadium (Chennai, India) and Eden Garden Stadium (Kolkata, India). The least amount of polygons to bring up the structures are used [13].

Figure 4: Visual research on the Stadiums



All-rounder: CSK

Figure 5: 3D model character1

Bowler: KKR

Figure 6: 3D model character2

Figure 7: 3D Environment

After completing the characters and models, all the assets were imported and tested in Unity [22]. The following ten images show the artefacts achieved during the 1st Quarter.

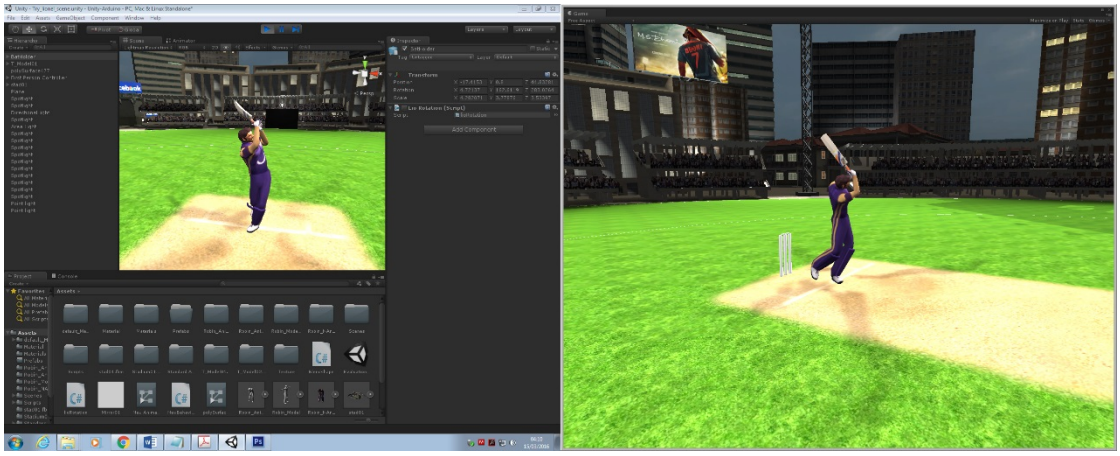


Figure 8: The Gameplay from Unity


facebook
facebook

Animation Artefacts

Motion capture

Motion capture is a technology where movements are tracked and recorded. This technique is used most commonly in computer graphics departments for movies and games [23]. In animation, motion capture helps to bring a new conventional method to make production quicker rather than the traditional moving and keying the character manually. Motion capture is also widely used in the field of physiotherapy and sports to render the solution for injuries or other complications [6].

It works based on the principle of tracking the motion using infra-red (IR) or passive reflection. IR devices such as a speed gun or thermal imaging camera could be the inspiration behind this technology. In motion capture technology, it is necessary to understand 'degrees of freedom" [24]. The term refers to the free movement of an object in 3D space. Achieving 6 degrees of freedom (6DOF) gives a complete motion-tracking experience because there are only six possible data required to determine free motion of an object in 3D space.

There are two primary methods existing [23] in the current state of the art to capture an action. On the researcher's view, they are active and passive means, but as per the industry terms, they are mentioned as:

- Optical tracking methods
- Non-optical tracking methods

In short, the optical tracking method is also said to be the imaging method that relies on a specialized camera. The non-optical tracking method relies on electronics and MEMS sensors to measure the data.

In some motion tracking, the technology uses both approaches for the benefit of more accuracy. For instance, the RVL-001 and RVL-101 Wii remote versions used the accelerometer with the IR tracking for the tilt. Then, later on, in "Wii remote motion plus" a gyroscope was added for more accuracy.

Optical tracking method

As mentioned above, the optical method of motion tracking uses a specialized camera to monitor the optical markers [6]. The visual markers are stitched onto the suit which an artist wears and performs in. Usually, the markers have a high reflective material that glows against the flash. Professionally, there are as many markers as possible attached to capture a perfect motion. The 41-maker set system is the best as far as this researcher knows. The system provides the possibilities to record all the body movements made by the biped (human) with details.

In gaming, it is enough to capture the motion using the 29-marker set system. The 29-marker set also does the work with the minimal effort [25]. In recent times, the cameras used for this are specialized to calculate the depth. The commercial models available for everyone in the market have fewer markers or no markers at all [2]. The model uses the current advancement of camera imaging technology, where a camera can image more than a flat picture.

Using fewer markers can reflect in a deficiency called marker swap [24] where the cameras involved become confused and view the same marker as two different markers which naturally results in an inaccurate tracking of the motion. On speaking about the markers, markers are separated into two categories [24]:

1. Passive Markers
2. Active Markers

The above has discussed passive markers. Active markers are just makers which are have self-powered LEDs instead of reflective material. These markers provide more accuracy than a passive marker and also have to be powered up in the suit all the time when it is used. The Xbox's motion controller remote with an LED tower is a prime example of this. Similarly, for passive markers, the Oculus Rift can be an excellent example which is available commercially.

Most of the commercial systems try to minimize the markers or adopt gesture-recognising technology. Kinect is one such innovation which can track six people

at the same time without any markers, but it cannot yield the same as professional equipment. Similarly, leap motion tries to recognize gestures by proximity. Even it also fails to achieve the outcome because the capture volume is very limited [6].

All the commercial systems in the current state of the art fail to get the full body tracked to the virtual world because of the limitations, and the games are meant to be played in a permanent location. However, this researcher is impressed with the Lighthouse tracking system, which will be available soon in the market [24]. The monitoring system uses a laser to scan the entire volume of the room and the character irrespective of obstacles. The idea behind the Lighthouse tracking is revolutionary, however the researcher has concerns about health and safety.

Non-optical tracking method

Non-optical method is the area chosen in this project. This method uses various sensors to achieve the result [26]. Since the advancement of electronics in the early 1990s, MEMS (Micro Electro Mechanical System) [5] are available which placed all sensors onto a small chip. The sensors – such as gyroscope, accelerometer and magnetometer – are enough to get the work done, as a gyroscope gives the rotation, the accelerometer gives the movement according to the acceleration and the magnetometer provides the orientation towards the earth's magnetic field. These sensors are powerful and have evolved along with the development of the mobile phone industry. The new upgrade sensor has exceptional accuracy with low latency [27]. A self-contained system without the optical techniques can be achievable in spite of many commercial systems combining both technologies together. These can be achieved through fusing the sensors by a sensor fusion algorithm which involves brilliant mathematics to make predictions based on the precision motion data [28].

Modern commercial products [24] such as Salto and GloveOne use flexi-sensors along with other sensors that generate data according to the threshold of the material which is bent. The Myo armband collects the data from muscle contraction and senses the movement about to be made and produces the track

accordingly. This works under a logic of recognising the gesture with the signals from the forearm muscles. The Myo also gives a clue for future innovation to determine the physical motion directly from the brain.

The mechanical method [26] [27], for instance, an exoskeleton tracking that involves many potentiometers, can also give an entire track. The Gypsy motion capture suite and Dexmo F2 are existing systems which work on this principle. Particularly, Dexmo F2 can provide force feedback, which is impressive.

Production

For the animations for the game, optical technology with passive markers is used in this project. The professional system which working with here was the Motion Analysis System [25]. The system comes with IR cameras that track the passive markers and also comes with an application called Cortex. Cortex version 1.0.0.198 is used here. The application runs with the emulating dongle that has the original licence in it. Eight cameras were wired into the capture system. The capture volume [24] is a rectangle cuboid with four cameras on both longer sides which are equally spaced and aligned facing down. These cameras were targeted towards the centre of the ground. Production is classified here into three categories:

1. Pre-production
2. Mid-production
3. Post-production

Pre-production

After categorising the scene, the script is developed with a storyboard [13] and the listings and props. The camera has to be switched on beforehand [25] because it takes time to heat up the IR cameras, and these cameras perform well 20–30 min after being switched on. Meanwhile, the cameras are adjusted according to the requirements of the scene. Then the capture volume is checked to make sure there are no reflecting objects in it. When the cameras are ready, the system is calibrated [24] [6] [23]. This entangles the cameras together

according to the new capture volume; the following procedures comprise the calibration:

- Launch the cortex software
- Load the project file. The project file holds calibration, threshold and masks, tracking parameters, marker sets and templates.
- Save the file progressively according to the date because the system needs to update the calibration on a daily basis.
- Avoid overwriting the files.
- Define the frame rate, shutter speed, brightness and threshold according to the need and connect the cameras.

System Calibration

The calibration must be done when the camera speed is 60 Hz.

L-Frame [25] calibration process steps:

- The L-frame has four passive markers on it. The L-frame needs to be placed at the centre of the capture volume.
- Check the orientations are right. In Cortex open the calibration tab from Tool>settings.
- Activate the cameras in the calibrated panel.
- The layouts in the menu give the possibilities of data views. Using the 2D view, inspect each and every camera. The camera can pick up signals from anything that reflect light. So those areas are to be masked before running the calibration.
- After setting the cameras to an optimized coverage, push the calibrate button and a sound is heard when the calibration is done.

T-Wand [25] calibration process steps:

- Then, remove the L-frame from the capture volume.
- Set calibration with the Wand field. Use the default wand length and adjust the capture duration to 60 seconds.

- An artist needs to cover all the possible areas in the capture volume using the wand after pressing the capture button.
- The movements can be the combination of horizontal and vertical waving and sweep.
- When the capturing is done a window pops up showing the status of the coverage called Calibration Status. Check there are as many markers detected by each camera. The average processed data should be above 500. If not, recalibrate again on changing the parameters.
- If it is 500 and above, then accept the calibration and save the project on the day's date and continue the production.

Mid-production

After calibrating the system, this system is now ready to track the markers. A test is run carrying a marker to the capture volume. The system should follow the marker alone. If it jitters or displays more markers, then the problem could be in any of the previous procedures. In this case, one of the cameras was found to have a deficiency. So that camera switched off and the capture volume realigned with the available functioning cameras.

After testing everything loaded, the marker was set to the project file. The "Helen Haynes Markers Set" was used. There are 29 Helen-Haynes [25] markers to place on the body.

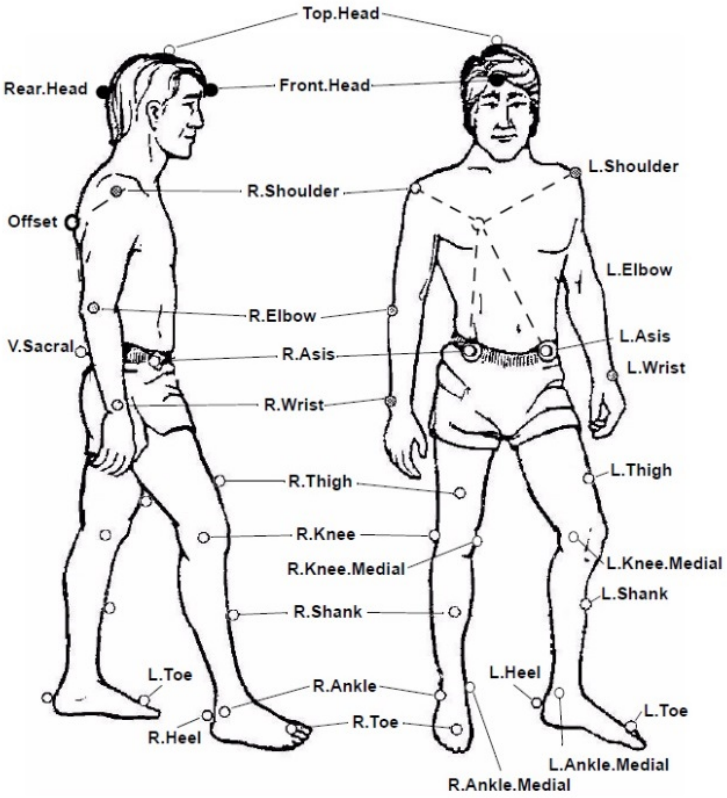


Figure 9: 29 Helen-Haynes Markers [25]

After placing the markers in the right position. To start capturing the scene, this researcher suggests capturing a scene to the maximum of 60 seconds per take. It could be good practice for a seamless workflow. The scene should start in a T-pose and must end in the same T-pose. A quick ID is needed to connect the cloud of points together. The motion files were exported in .trb or .trc format. Before consigning make sure that the files are 100% cleaned. Cleaning is the primary task of the production. It involves various tools to fix the curves. The curves should be smooth, jitter-free and complete. Some of the necessary tools are Rectify, Template Rectify, Cubic Joints, Virtual Joint, Smooth, Cut, etc.

A perfect capture should require less cleaning. The cameras should be calibrated correctly. When there is some deficiency in the initial steps, the camera behaves strangely, causing major marker swap, which is too difficult to sort and takes more time to clean up.

After the raw data are cleaned up, the results can be stored on flash drives and removed from the system. The raw data, which is the cloud of dots, are processed and converted into an animating skeleton with the help of a plugin called Calcium in the post-production.

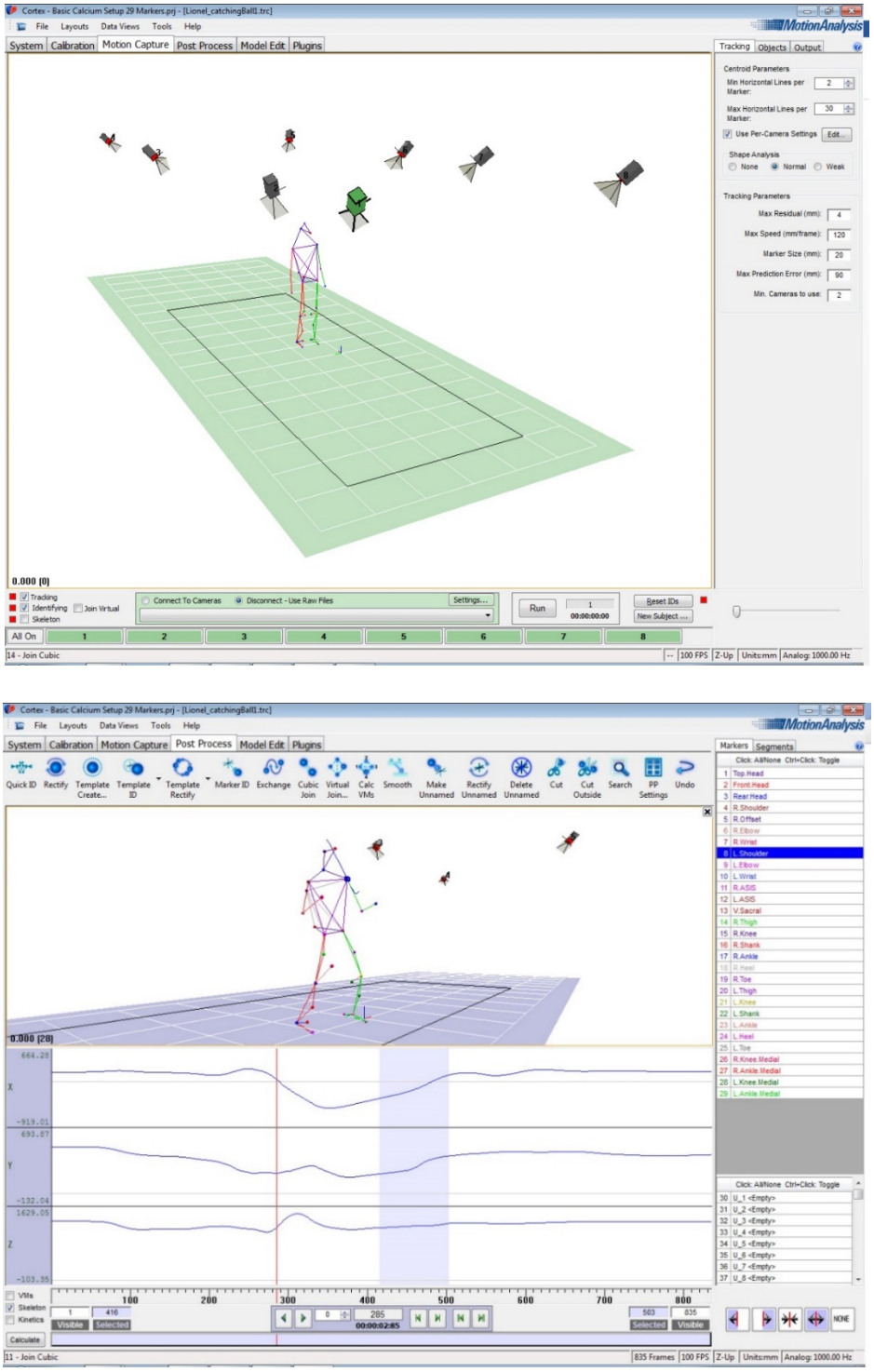


Figure 10: Motion Analysis GUI

An important note is a need to prepare a new Calcium bone for a different artist because the anatomy differs from person to person. Calcium will give a good result when the markers are the placed in the right position. So it is desirable to have a suit with markers stitched on or a suit with Velcro to attach them. In this case, using double-sided tape with the markers caused problems. Then after analysing the mistakes, the areas were marked on the suit where a marker was needed to be placed precisely for the production.

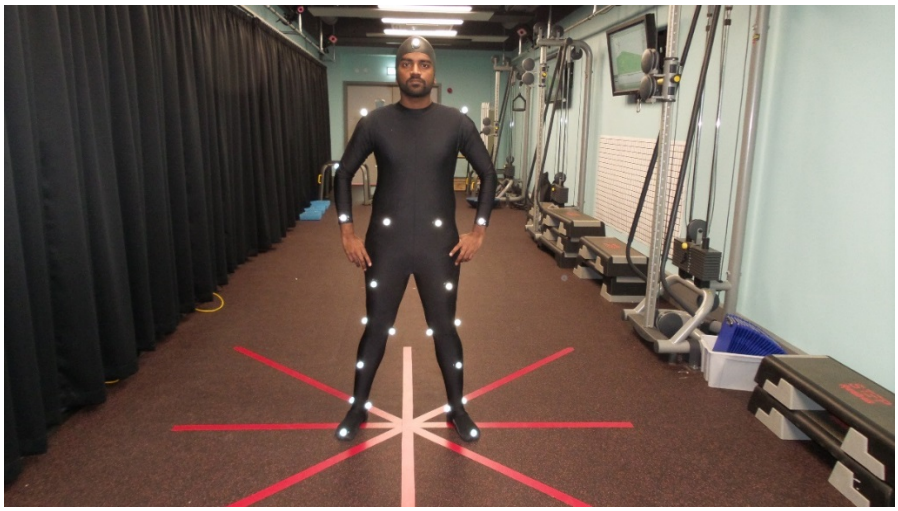


Figure 11: Motion Capture Production

Post-production

Calcium is used to solve a skeleton according to the marker data. As mentioned earlier, it is a plugin and works along with the motion analysis software. This needs to follow these steps:

- Load the project. Import a pose base skeleton. The skeleton is a .htr file and the skeleton functions like a manikin doll. Hence, it is possible to set a pose moving the joints.
- After importing, adjust the base pose skeleton such that it gets fitted into the cloud of markers set.
- Analyse the bones so they are centred to the markers' volume, both inside view and top view, and adjust accordingly.
- All the movements were manipulated interactively by changing the values in the menu box.

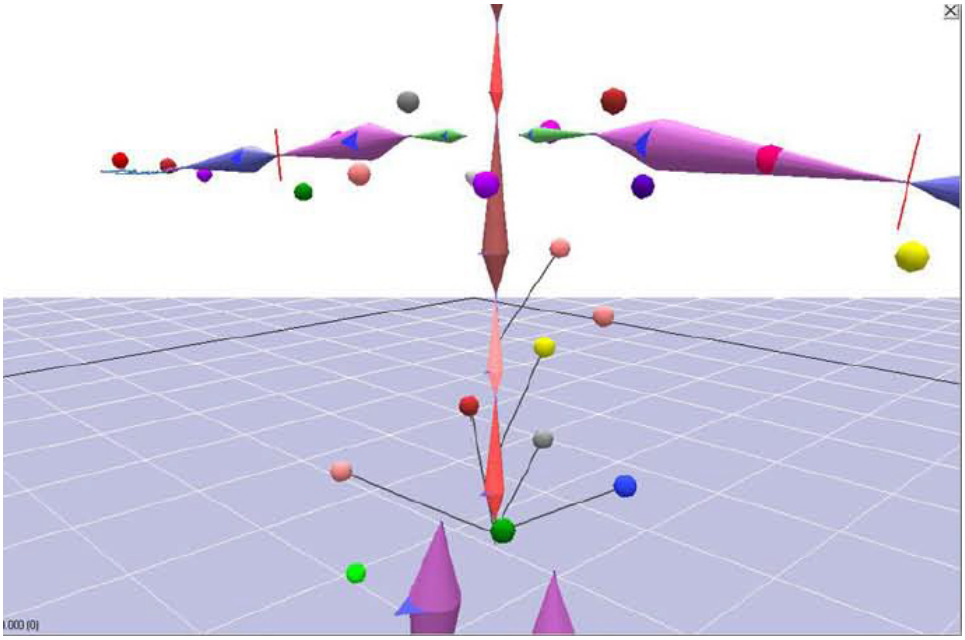
- Check the values in the global scale. The values should be 0.97.
- Save the project periodically because the system will not allow undoing a mistake.
- The next step is to assign degrees of freedom to the bones. Starting with the hips, assign the levels according to the movement a joint is capable of.

For instance, a hip is capable of all possible actions, hence it gets assigned to 6DOF Global. Whereas, the chest bone is given 3DOF spherical. These are done interactively in the menus on selecting the particular joint.

Segment Name	Joint Type
Hips	6 DOF - GLOBAL
Chest	Master Segment - Chest3
Chest2	Master Segment - Chest3
Chest3	3 DOF - Spherical
LeftCollar	2 DOF - Universal
LeftUpArm	3 DOF - Gimbal
LeftLowArm	1 DOF - Hinge
LeftHand	3 DOF - Gimbal
Neck	Master Segment - Head
Head	3 DOF - Spherical
RightCollar	2 DOF - Universal
RightUpArm	3 DOF - Gimbal
RightLowArm	1 DOF - Hinge
RightHand	3 DOF - Gimbal
LeftHip	3 DOF - Gimbal
LeftLowLeg	1 DOF - Hinge
LeftFoot	3 DOF - Gimbal
LeftToe	1 DOF - Hinge
RightHip	3 DOF - Gimbal
RightLowLeg	1 DOF - Hinge
RightFoot	3 DOF - Gimbal
RightToe	1 DOF - Hinge

Figure 12: Table for DOF specification [25]

- The next step is to attach the markers to the driving bone. The attachment was done quickly by clicking the bone. In the menu, click “attachment”, which toggles software into the attachment mode then left click the corresponding markers to attach. A connecting black line will emerge from the bone stating the connection was made.



Segment	Markers to Attach
Hips	LFrontWaist
	LBackWaist
	Root
	RBackWaist
	RFrontWaist
Chest	Root
	LowerBack
Chest2	LowerBack
	MiddleBack
Chest3	LSternum
	LSShoulder
	TopSpine
	RSternum
	RSShoulder
LeftCollar	LSternum
	LShoulder
	LSShoulder
LeftUpArm	LBicep
	LOuterElbow
LeftLowArm	LWrist
LeftHand	LT_Thumb
	LT_Pinky
	LWrist
Neck	TopSpine
Head	LFrontHead
	LBackHead
	RBackHead
	Head_Asym
	RFrontHead
RightCollar	RShoulder
	RSShoulder
	RSternum
RightUpArm	RBicep
	ROuterElbow
RightLowArm	RWrist
RightHand	RT_Pinky
	RT_Thumb
	RWrist
LeftHip	LThigh

Segment	Markers to Attach
LeftHip	LOuterKnee
LeftLowLeg	LAnkle
LeftFoot	LHeel
	LOuterMeta
	LInnerMeta
	LToe
LeftToe	LToe
RightHip	RThigh
	ROuterKnee
RightLowLeg	Rankle
RightFoot	RHeel
	ROuterMeta
	RInnerMeta
	RToe
RightToe	RToe

4. Save your Project.

Figure 13: Characterising the Markers

- The last step in this procedure is to add weights to the markers. Categorize the weights into 1, 2 and 3.
- Do this in the Tree view tab under the Calcium segments.
- Select the joints and browse further to locate the marker and enter the values for the property weight. The values are:

Segment	Marker	Weight
Hips	LFrontWaist LBackWaist Root RBackWaist RFrontWaist	3
LeftCollar	LSShoulder LSternum LShoulder	2
LeftHand	LWrist LT_Pinky LT_Thumb	2
RightCollar	RSShoulder RSternum RShoulder	2
RightHand	RWrist RT_Pinky RT_Thumb	2
LeftFoot	LHeel LInnerMeta LOuterMeta LToe	3
RightFoot	RHeel RInnerMeta ROuterMeta RToe	3

Figure 14: Adding Weights to the Bones

- After finishing all the steps, solve the skeleton by clicking the bone symbol. The animating bones can be seen when playing it.

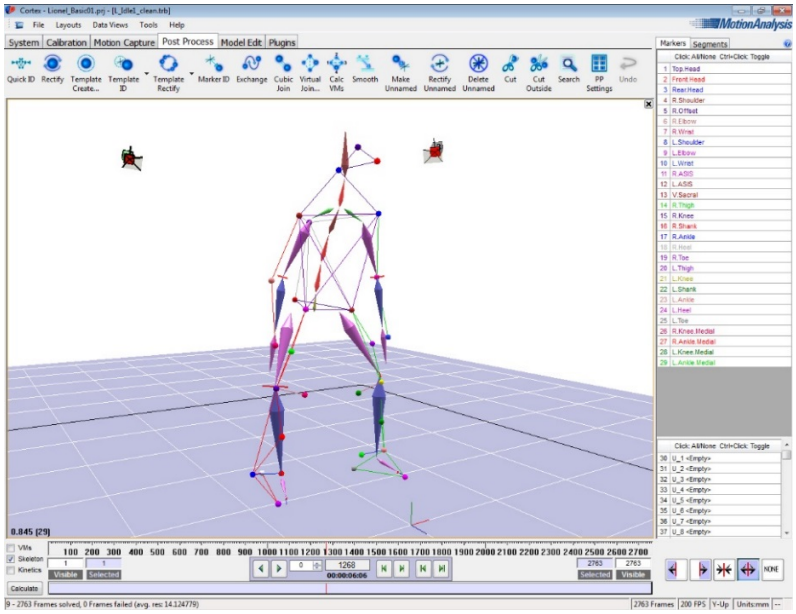


Figure 15: Motion Analysis GUI with the Solved Skeleton

- Then the animating .htr bones can be exported from the motion analysis software to any third-party 3D software. The .htr is a standard format for the motion files.

Thus, this was how the production was made with the Motion Analysis System, which is one of the professional motion tracking systems used in the industry.

Motion analysis of Unity pipeline

Here the pipeline [23] followed to incorporate the animation made using the motion capture for the game is discussed; this is the final part of the artefacts. The Motion Analysis System was used to capture the movement and Calcium to solve the animation bones. The animating bones are now in the .htr format.

The software used to produce the animation in the game is as follows, and is discussed in the following sub-sections:

- MotionBuilder [30]
- Human IK System Maya [15]

MotionBuilder

MotionBuilder is a powerful tool for doing real-time 3D animation, but this is not a complete 3D content creator like Autodesk Maya and Max. Even capturing real-

time motion is possible using the motion builder under the Autodesk forum. MotionBuilder is also suitable for pipelining all 3D content creator.

Most of the animating tools in MotionBuilder are similar to Maya except some motion capturing tools and options. In Maya, the category under Animation holds most of the tools which are in MotionBuilder, such as dope sheet, graph editor, etc. Even though the researcher knew how to use this tool, MotionBuilder was used initially only for editing an animation and exporting it to a .fbx file. The .fbx is a common file format for Maya, Max and Unity.

In the production pipeline, MotionBuilder was used for a very limited purpose because this researcher created much more of the work in Maya. More than 80% of the significant features of MotionBuilder were incorporated in Maya 2015 under the animation category. This tool is used for:

- Creating a control rig
- Loading the animation
- Editing non-destructively
- Retargeting the animation
- Importing and exporting the animation

Creating a control rig

After modelling and skinning the character through a 3D content creator, the file was imported to MotionBuilder. Then a set of procedures is followed to add the “effectors”, which are manipulators which allow posing the character. All the effectors collectively give total control of the biped and this is called a control rig.

Loading the animation

MotionBuilder allows loading the animation irrespective of the format, such as .bvh, .htr etc. The software provides a broad spectrum; it even allows to pipeline third-party software. Here, the Motion Analysis System was used to generate the animation. The loading of an animation is as simple as loading a file.

Editing non-destructively

MotionBuilder has a multiple-layer editor for animation. Hence, a layer can be added on top of it. These layers allow to by-pass or alter the animations non-destructively.

Retargeting the animation

Regardless of the anatomy or scale, MotionBuilder tool enables copying an animation from one biped to another biped skeleton. The adaptation is a crucial method for doing an animation using the imported motion-captured animating skeleton. Here, the model with the control rig can be retargeted from the imported motion file.

Importing and exporting the animation

After tweaking the animation, the animating key frames are baked into either the control rig or the skeleton. Then the model can be exported with the animation. Similarly, a model can also be imported with the animation so that it can be edited non-destructively or can be a source for retargeting another model.

Graphical User Interface

There are many ways this tool can be used. The tool has a graphics user interface (GUI) which is simple in the style of the editor.

- It has the **General menus**. In menus, the **help**> view topics contributes to research in what the tool is capable of.
- Here the interaction mode can be changed following the settings> interaction mode>Maya. This researcher's preference is for Maya''s interaction mode to navigate and its shortcut keys.
- In the menu, the number of backup files and Undos were specified using the setting preference.
- In **layout,** toggle the modes like editing, scripting and preview screen can be toggled.

- Apart from the menu, the **ports and displays** are similar to Maya.
- The **Transport Controls window** is similar to a movie editor with takes, timeline, scrubbing, zooming, play, pause, stop, play per frame, speed up the frame rate, etc.
- The **Resource window** is the explorer for the file, folder and various other resources.
- The **Character controls window** is where the control rig effectors were built, and the motion tweaking process was done through a process called **characterisation**.
- The **Key controls window** allows editing the animation manually by adding and removing the keys. The method also helps to animate a character manually in the traditional way.
- **Animation Layers Window** contributes to modifying the animation by adding a layer on the top of the pose base animation.
- **Navigator Window** is similar to the Outliner in Maya and also leads to the attribute editor to change the quality of the selected in the scene. The Navigator window also comes with animation control tools like **Dope sheet**, **Function Curves,** which are in Maya as well. The **Story** is a powerful tool for visualizing the animation scenes with multiple cameras. This tool also creates animation cycles and blends them. The **Animation trigger** tool helps animation creation using the event from the gamepad, mouse and keyboard interactively.

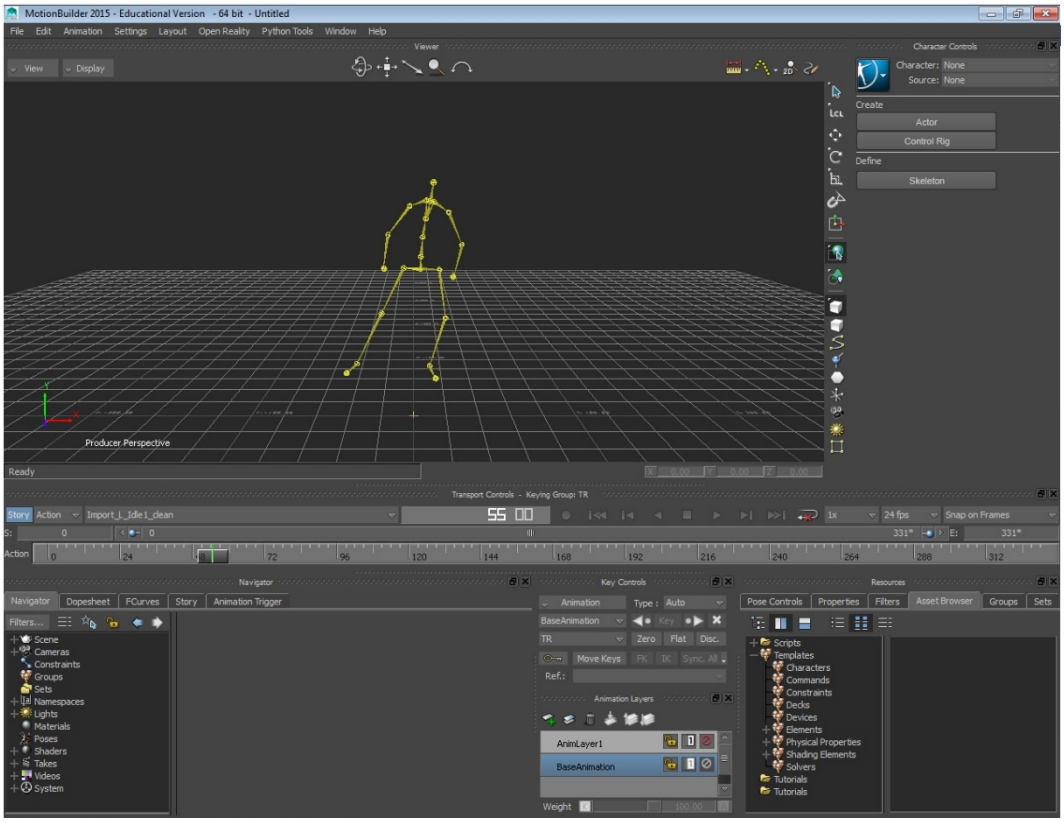


Figure 16: Motion Analysis System GUI

Human IK System (Maya)

Human IK System [15] is a powerful tool in Maya for characterising the animation. Maya was used to pipeline between motion analysis and Unity, even though using MotionBuilder is enough to pipeline both. As a 3D content creator, Maya has the tools necessary to do this process. So there is no compulsion to stay in MotionBuilder.

After converting the animation into .fbx file import in Maya, it was stored as a Maya file, either binary or an ASCII file [14]. A folder was used to organize all the takes and moves. A new project was opened in Maya and the model imported, which should be modelled in a T-pose.

Figure 17: The character "Robin" in T-Pose

For example, take the character "Robin" above, a wicket-keeping batsman. After importing him, the next process is to skin the character. The skinning is a simple but time-consuming process where the skeleton is created and attached to the model [21]. Before skinning, the models must be clean from history.

Skinning and blend shapes

Skinning [16] involves methods to determine the influence of the bones on the model's mesh. This was done by selecting the bones and applying weights to the mesh accordingly. Weights were added in the process of painting on the mesh [14], the tool is called Paint Skin Weights. Weights were painted after bonding the skin to the model's mesh. For a good result, the model was posed in all possible ways and painted. The process removes the errors were there was deformation. On using the tools under Paint Skin Weight, the deformations are adjusted to perfection. The example model is primarily made up of two objects, namely the body and head. A smooth bind to the body and a rigid bind to the head were made because the head should not be deformed in the course of the

animation. Another method was used to animate the face expression, namely “Blend Shapes” [15].

Blend shapes is another technique in the animation for animating an object using the component mode [14]. A **vertex** is a component that connects the polygons’ edges. This blend shape tool allows and controls the vertex animation. It was done by the following steps [15].

- An object was duplicated from the source object. In this case, the source object is Robin’s head.
- In the duplicated objects the vertices are adjusted to make a pose such as opening the mouth or closing the eyes.
- Then the copied object’s animation is retargeted to the source object and keyed.

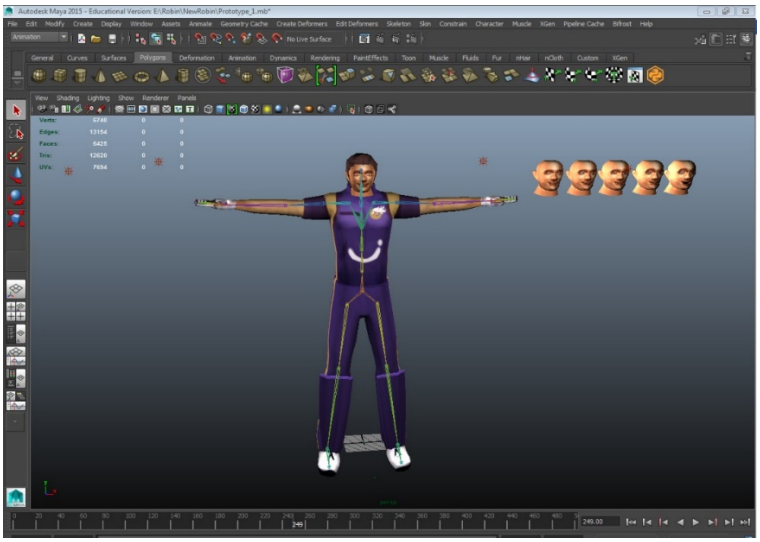


Figure 18: Model with Blend Shapes

After skinning is done, the next process is to rig the character. The traditional approach to rigging a character was followed here. The introduction of the Human IK System [14] in Maya has made the work even simpler. It is the character control window from MotionBuilder. Using this powerful tool, kinematics and constraints can be established instantly and easily without manual effort while rigging.

The Human IK tool

Human IK System [15] needs the skinned biped model to be characterised to perform the operation. Characterisation is a process where effectors can be added to the biped. The effectors combined form the control rig for the biped.

This process is done easily when the character model is in the T-pose. The system recognises the biped only when it is in the T-pose. In this process, select a bone from the model and map it to the Human IK System's character correspondingly. The process is started at Define > skeleton. This needs to be done for each and every bone to relate the model to the Human IK System. After completing the process, the bones are all checked for mapping in green in the Human IK window. Where not, the joints are adjusted such that the map turns green when the joints are in the proper position. The process is completed when all joints are in the correct position. At this point, the file is saved to acknowledge the successful characterisation [21].

Creating the effectors starts at create > control rig. It will create the effectors instantly forming the control rig: an instant way of rigging a character. It generates the whole body kinematic system and also defines the gravity and constraints related to it. [14]

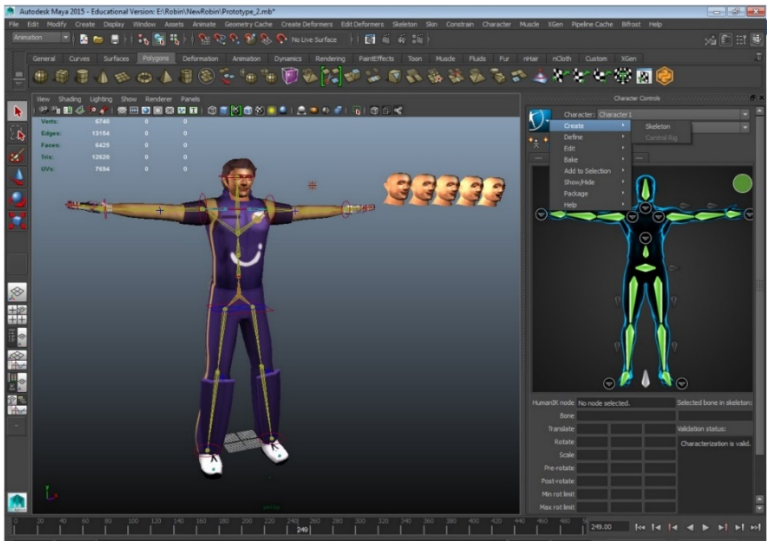


Figure 19: Model with Human IK Characterisation

The circles on the joints in the figure are the effectors. These can be moved and keyed. They manipulate the character to achieve the desired pose very quickly [15].

The model must be characterised to the corresponding joint in the Human IK System. The listing below shows the characterisation.

Note: The video of the process has been uploaded to the internet in the following link: https://vimeo.com/146193090

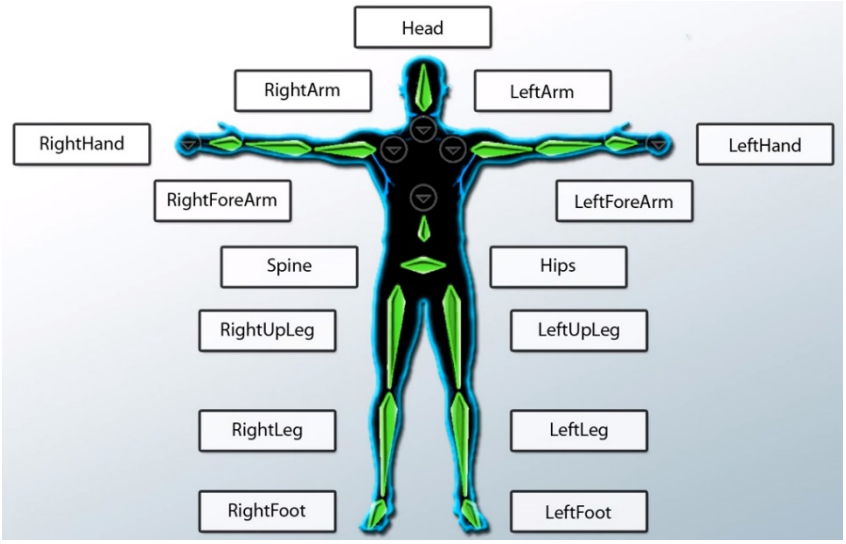


Figure 20: Characterisation Chart

Similarly, there is a process for a Custom Rig. The custom rig can bring customised controls according to requirements. This researcher's preference is to make a default control rig through a Human IK System [15].

Tweaking the animations

Retargeting the animations is a feature in the Human IK System, and is the most crucial for the pipeline in doing an animation using motion capture data. Here, the animation from the motion file should be retargeted to the model of Robin.

The Robin model is now characterised and saved. Now the Maya motion files stored in the Animation directory are imported, such as an animation like slogging the ball. Next the animation is characterised. The animation was captured starting and ending in a T-pose.

If the animation does not have a T-pose to start or end with, then it needs to be brought to a T-pose and then characterised. This can be a time-consuming or even an impossible task. Later on, this researcher found a trick for playing with the tools.

- Select the pelvis of the animating skeleton such that all the bones are selected.
- Then go to Skin > go to Bind Pose [14].

It brings the pose to the T instantly. The animating skeleton can now be characterised as for the model Robin. The characterisation gets done by defining the skeleton to the Human IK System.

Within the character control window [30], the slogging animation control rig is assigned as the source and the model Robin control rig as the character. It connects both the control rigs together and entangles them. This process is called re-targeting the animations. Now when play is pushed, the model Robin animates, tweaking the signals from the source animation. To get the keyframes, baking process is used within the character control window [30]. The keys can be generated either to the bones or the control rig.

After getting the keys baked, the keys can be modified non-destructively by animating a layer over the top of the source layer.

After editing the animation now, the result is exported as an .fbx file, which is the primary asset data format for Unity in games development. Thus, this is how a pipeline between the Motion Analysis System and the game engine Unity is achieved.

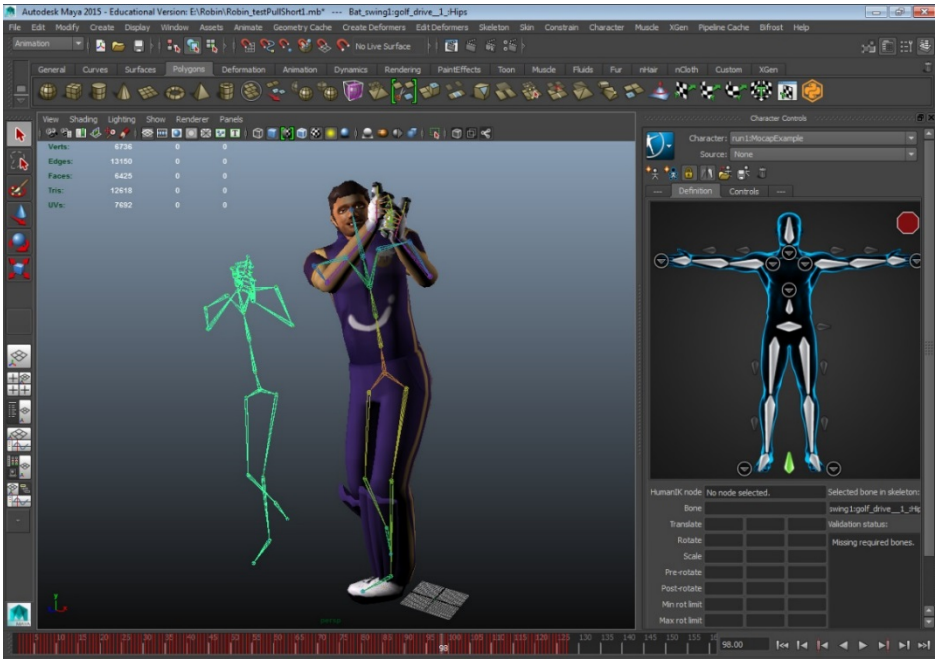


Figure 21: Model Retargeting the Animation

Comparison of animation with motion capture

The data were collected from a teaching class organised by the researcher's supervisor. The master class was about animation using motion capture. The evaluation is a short survey on the comparison between the methods.

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following data will help to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and also gives the quality in animation.

proposed Pipeline:

pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

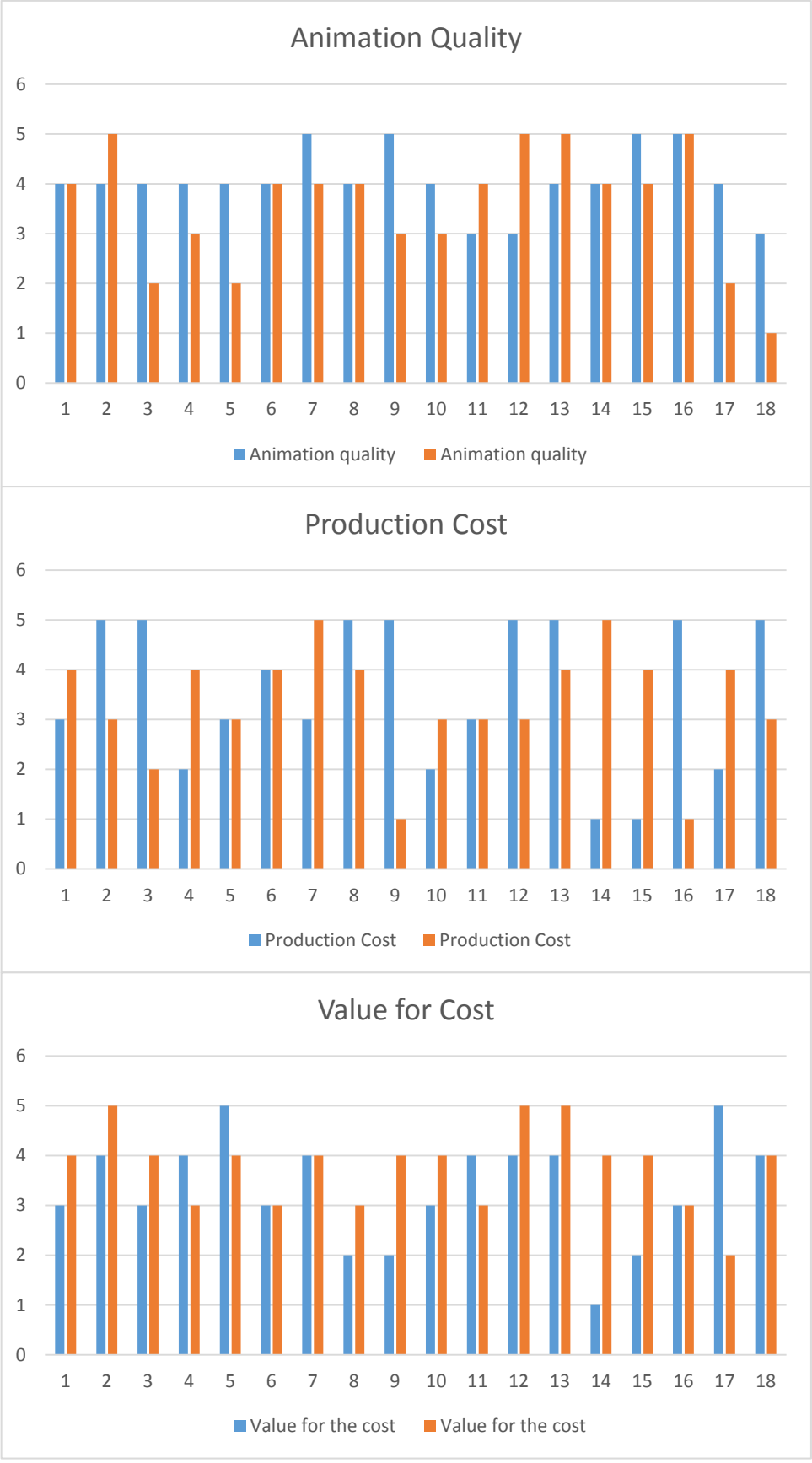
Evaluation : Please mark the columns from the scale of 5

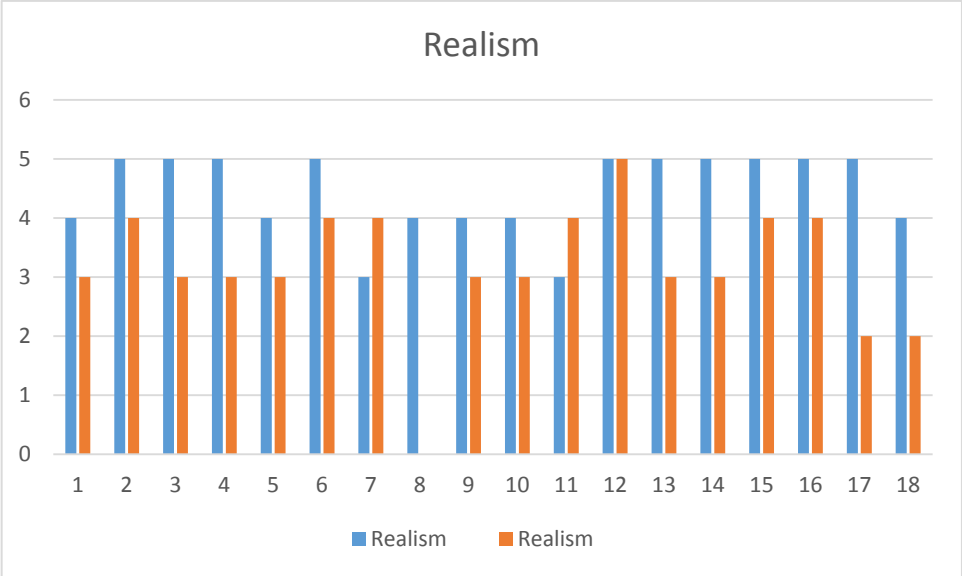
	Pipeline 1	pipeline 2
Animation quality		
production cost		
value for the cost		
Utilization of time in production		
Accuracy		
Realism		
Utilzation of creative skills		
Utilization of efforts		
Maintainance		
recomendition		

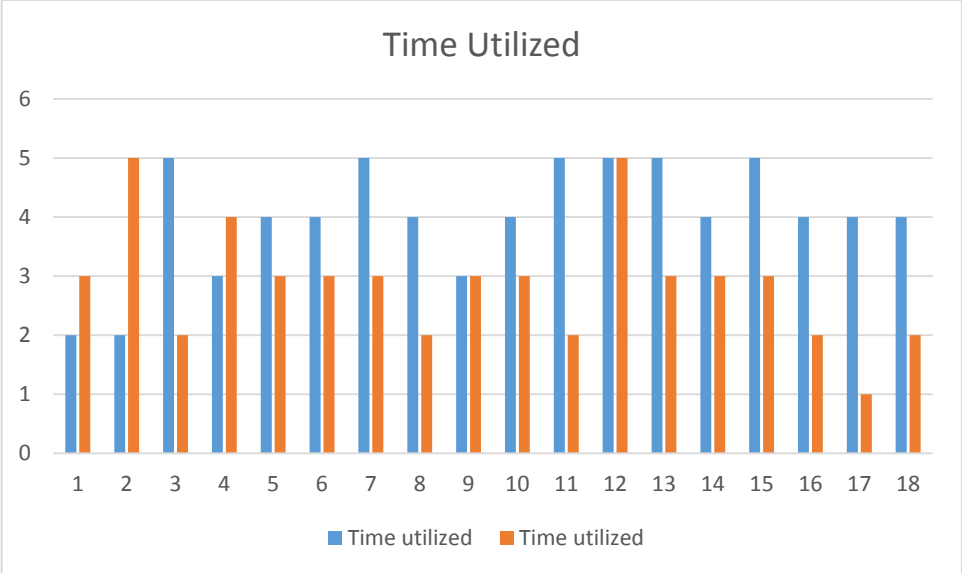
Please fill your details:
Name:
Area of Studies:
University:
Signature:
Your comments on commercial motcap system in market:

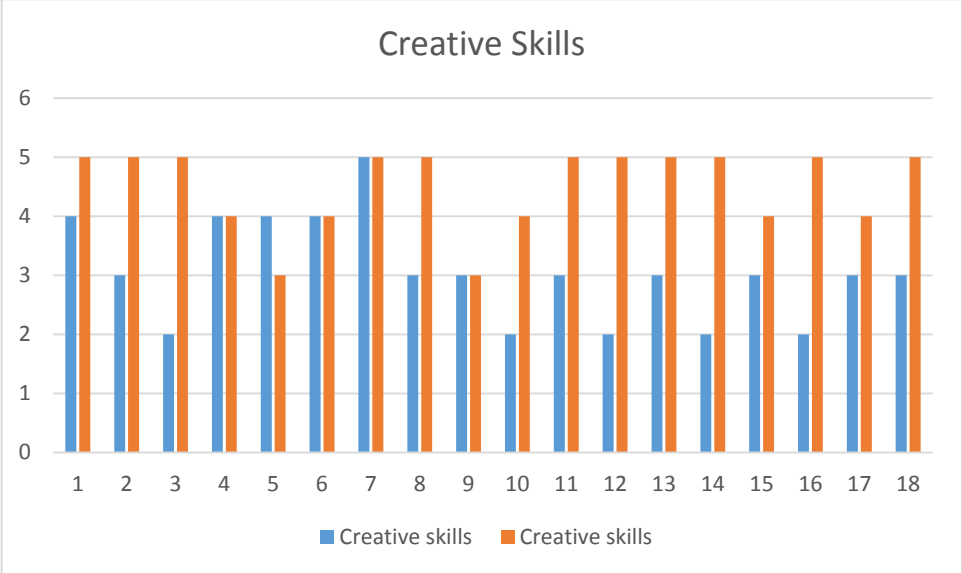
Figure 22: Evaluation 1

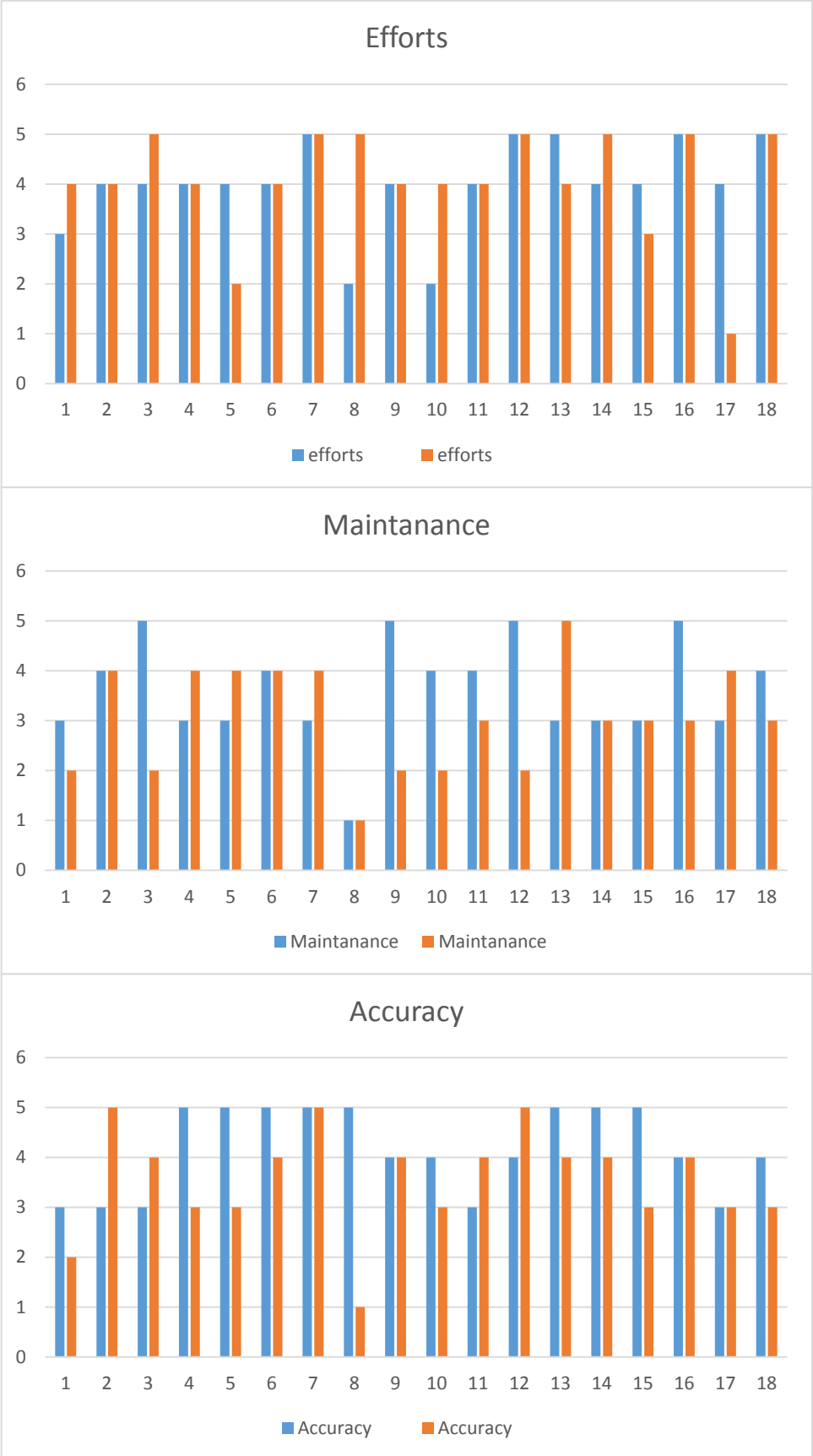
Figure 23: Master Class on Motion Capture


Animation Quality
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Animation quality
Animation quality
Production Cost
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Production Cost
Production Cost
Value for Cost
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Value for the cost
Value for the cost


Realism
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Realism
Realism


Time Utilized
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Time utilized
Time utilized


Creative Skills
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Creative skills
Creative skills


Efforts
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
efforts
efforts
Maintanance
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Maintanance
Maintanance
Accuracy
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Accuracy
Accuracy

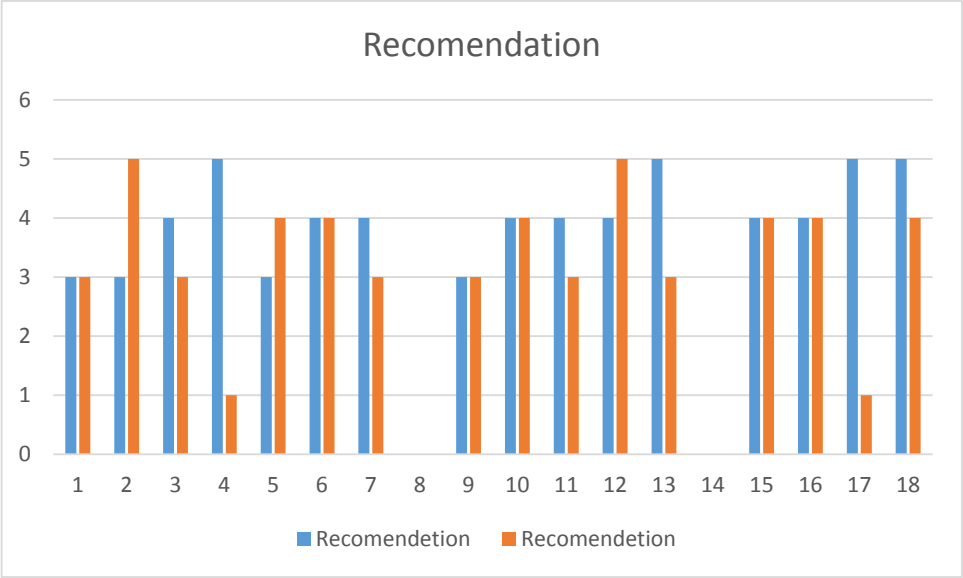
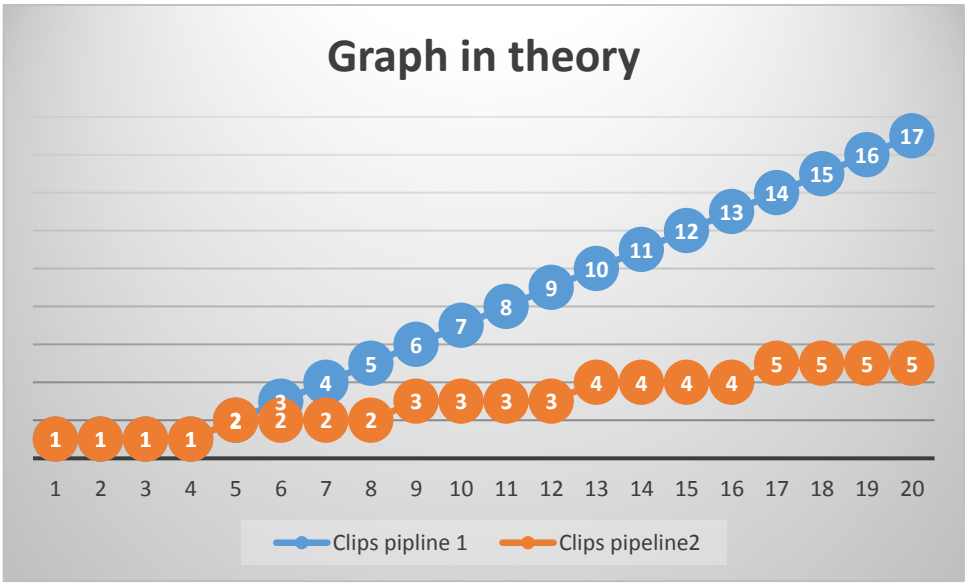


Figure 24: Charts for Evaluation 1

These data were gathered from the animation, creative technologies, and games development students. While doing this, they were given an outline of the SME (Small and Medium Enterprise). As per this researcher's knowledge, all the questions are hypothetical regarding an SME. The pipeline should fit for a bigger organisation to reduce the workload. The researcher created a theoretical graph that shows the exponential number of clips generated in a given time. This shows that the process reduces the time invested to a quarter of an hour per clip apart from the hour spent on the first clip.



The graph shows the clips generated were plotted against the time. It shows, in pipeline one, 17 animation clips could be generated in 20 hours whereas the traditional way could produce 5 clips in the same time. The overall evaluation shows a mixed result from the students and difficulties to choose the best.

Then the researcher developed an IMU-based motion tracker which tracks the motion passively without usage of the optical cameras. A further survey was done for studying the comparison between the optical and the non-optical method. The motion controller developed was tested with the students participating in the evaluation.

Comparison of the optical and non-optical motion capture

The data were collected involving students from the university. The same set of questions was asked, but the study is a comparison between the optical method and a non-optical method for a SME.

Since they had participated in the earlier evaluation and the master class on “Motion Capture Technology” the questions are formed to evaluate their beliefs.

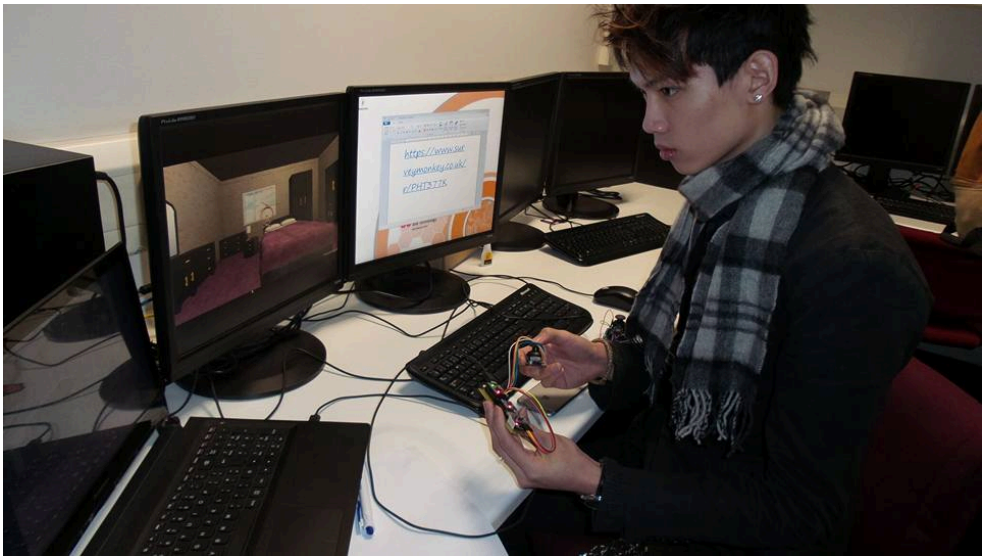


Figure 25: Testing the Motion Controller

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The caliberation is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	**pipeline 2**
production cost		
Utilization of time in production		
Accuracy		
Maintainance		

Please fill your details:
Name:
Area of Studies:
University:
Signature:

Your comments on commercial motcap system in market:

Figure 26: Evaluation 2

The graphs are shown below.

According to the knowledge of the people surveyed:

- Most participants felt the optical method would be high cost.

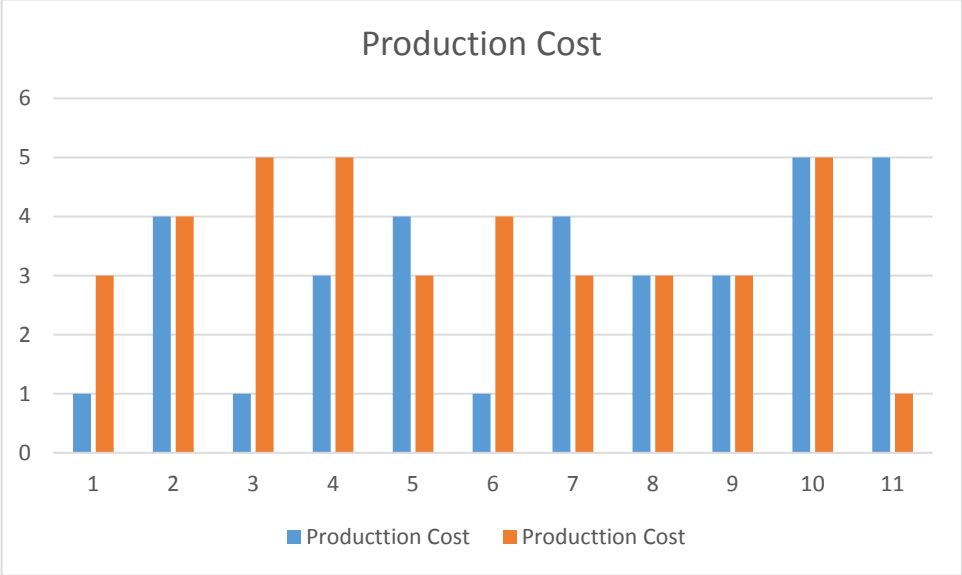


Figure 27: Evaluation 2 Production Cost

- The accuracy of the optical method should be the best because the optical method does not need any compensation or estimation. Everyone guessed that optical method should be accurate on knowing about Oculus Rift [1], which is a commercial tracking device using the optical tracking approach.

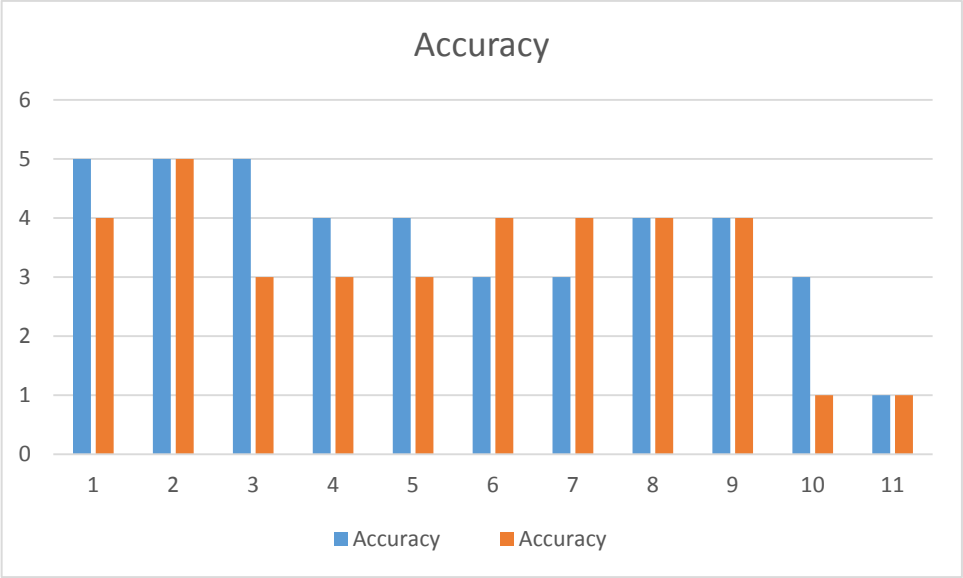


Figure 28: Evaluation 2 Accuracy

- The majority of the participants were also able to predict the utilization of time in production. This is because most of them tried the motion capture facility at the university. They also know the factor of the expensive camera which needs high maintenance.

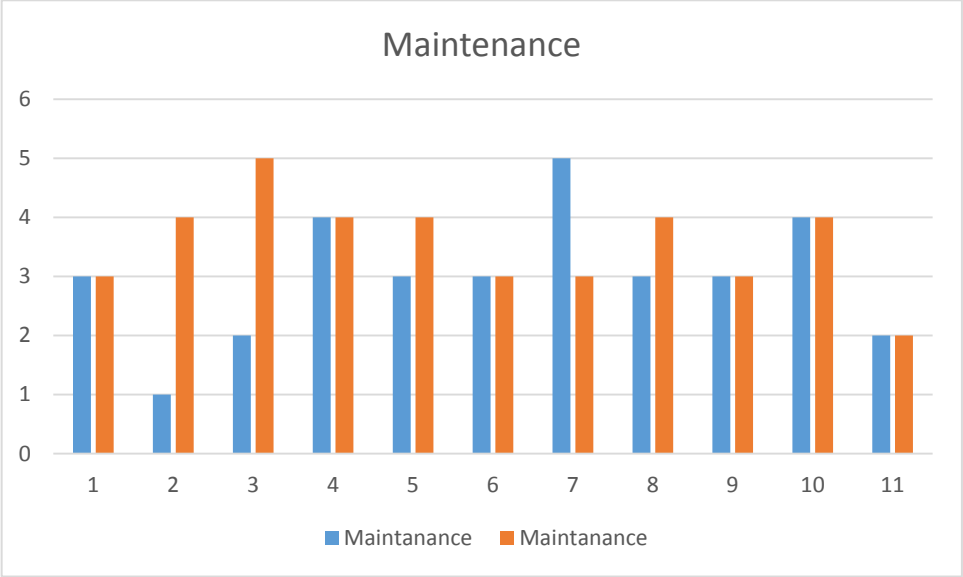


Figure 29: Evaluation 2 Maintenance

Then the focus turned to analysis and study of the motion capturing technology.

Electronic Sensors

Introduction

First and foremost conducting a study on developing a motion controller [5] for video games requires a deep understanding of the sensors which compliment tracking the movements in the 3D space [26]. For instance, a “barometer” is one such sensor which can calculate the altitude in relation to pressure, which decreases as the height increases. Similarly, many types of sensors have different methods for the same purpose.

It is all about choosing the sensors wisely for the project. Motion controllers for video games do not require a barometer, whereas tracking the altitude in a flying drone will need one of these.

The iPhone 4 technology [38] has introduced a component called a “gyroscope” [26], which is installed at the centre of the every iPhone manufactured since. The parts contain sensors like accelerometer, gyrometer, compass, proximity sensor, which allow people to play games interactively as they move their phone. The VR cricket game which would be developed here needs the measurements from the accelerometer, gyrometer, and compass to calculate the position of the moving bat to input into the game.

How to get the input from a free moving body into the 3D world requires investigation of various research [1] [3]. The history of the Wii Remote provides an idea: the earlier version had an accelerometer and used IR for tracking the tilt. Now the current version has upgraded to a gyro-based motion controller. As the result, the conclusion here was that the accelerometer and gyrometer are the essential components where the focus should be to get the acceleration and rotation (yaw, pitch, and roll) [37].

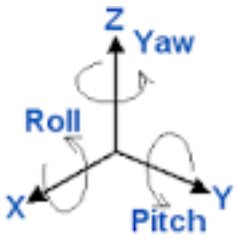


Figure 30: Yaw, Pitch and Roll

Also, a magnetic compass helps to get the orientation corrected by the earth's magnetic field.

An ideal Inertial Measurement Unit (IMU) needs all the three primary sensors to capture the motion passively [5]. The sensors need a programming board to run the program [31]. At the same time, it also needs to activate the circuit to get the functionality. Arduino is one of the best available open source programming boards. Uno boards were used initially; later Leonardo [32] was found to be the kind of board to be chosen because of its ability to connect to any device as external hardware, such as a mouse or keyboard.

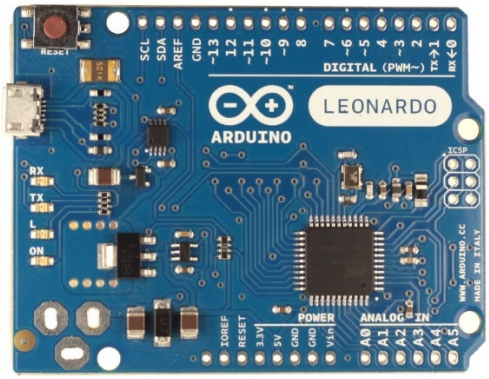


Figure 31: Arduino Leonardo for testing

The Arduino Leonardo was based on ATmega32u4 and combines both analog and digital pins with 5V; impressively it has micro USB 2.0 which is found in many mobile phones [31].

The same functionality is found stripped down into smaller boards such as the Arduino Micro and Arduino Pro Micro for handling convenience [32]. These boards have local registers for a particular purpose. The idea was to use the

bigger board for prototyping and testing. The smaller boards were used for implementing the VR goggle and the game controller.

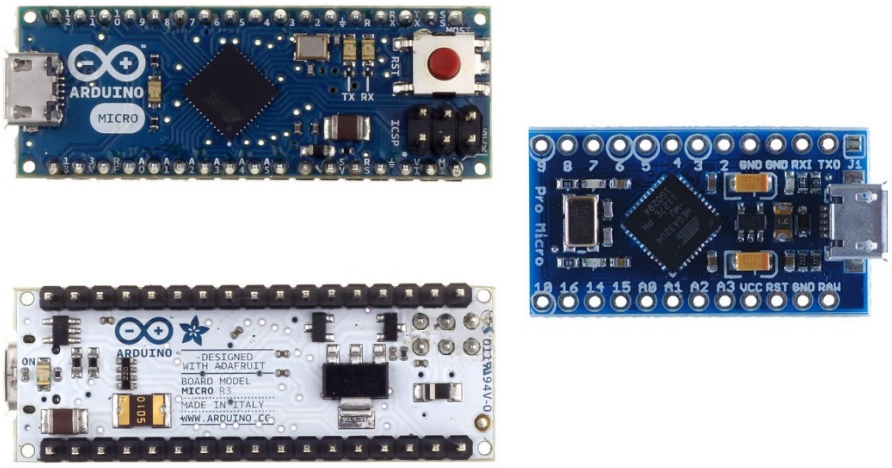


Figure 32: Arduino Pro Micro and Mini

Before proceeding with the project, a study was made to understand the logic behind the sensors being used.

Magnetometer

The magnetometer [39] has been widely used for navigation purposes in military, aerospace and industrial arenas. From the year 2000, electronics have emerged to make it smaller to fit in appliances using MEMS [5]. The magnetometer uses methods like the Hall Effect which uses the principles of magnetic induction, GMR/MTJ (Giant Magneto-Resistance / Magnetic Tunnelling Junction) electronic quantum method of measuring the variation in voltage caused by electron movement, AMR (Anisotropic Magnetoresistance) which uses a metal called permalloy, and the recent one uses the 5th Lorentz force.

The electronic magnetometer miniatures are also commonly called e-compass gauss meters. These are available in two axes and three axes, as well. The three-way compass gives the orientation no matter how is held. Some are available in combination with an accelerometer for the orientation [5].

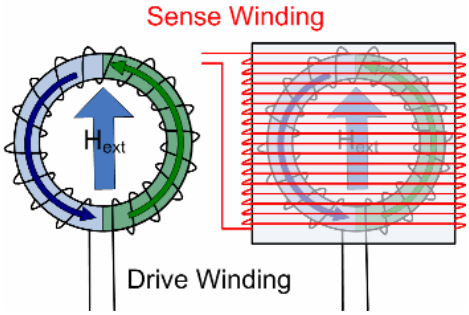


Figure 33: Working Example for 1-Axis Magnetic Compass

The major drawbacks of magnetometers [27] are magnetic inclination and magnetic declination. For instance, a magnetic compass placed on the North Pole of the earth should need a GPS (Global Positioning System) for the adjustment mean to be made in correcting the errors due to the surface of the earth, which is perpendicular at the North Pole.

However, the drawbacks will not have an impact on the device unless the instrument was used in the polar region.

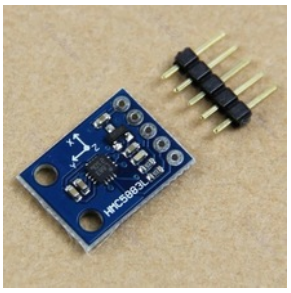


Figure 34: MEMS Tri-axis Magnetometer

The sensor employed in the project is HMC5883L, which works with 3 to 5V and uses I^2C bus interface with a range of ± 1.3-8.0 Gauss.

Accelerometer

The accelerometer [40] is a device used to measure the acceleration [5] or the vibration of a body which is disturbed. It works under the principle of Newton's laws of gravity and the variation in the measured voltage due to gravity and calculates the acceleration.

Acceleration is the rate of change in velocity with time. The fundamental equation for the accelerometer can be written as

$$a(t) = -\frac{k}{m}x(t)$$

The logical model of an accelerometer is a ball of mass *m* tied at opposite ends to two springs such that each spring ideally at rest should be of length *k*. If, the ball is accelerated the *k* get shifted into *k-x* or *k+x* based on the direction of acceleration.

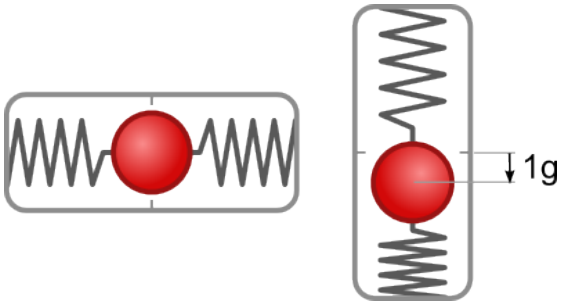


Figure 35: Working Example for 1 Axis Accelerometer

In a miniaturized MEMS electronic accelerometer [5], the spring and ball are replaced by fixed electrodes to calculate the variation in voltage due to the subtle deviation of the anchor in between them.

The disadvantage of the accelerometer [27] is the linear acceleration. The linear acceleration will add more noise to the accelerometer. It even amplifies small jerks and jitter. Gaming uses a low-pass filter to remove the jitter, which is why accelerometer-controlled games obviously have some delay, and the game is slow to control due to the implementation of the low-pass filter.

However, implementing the accelerometer in the project gives the advantage of calculating factors including the earth's gravity. The MEMS accelerometer is available in 2-axis or 3-axis or combination with gyrometer or magnetometer. It is wiser to use the sensors based on the purpose of the project. The sensor in this project does also have a gyro, so it gives 6 degrees of freedom [33].

The sensor used in the project is MPU-6050 6DOF, works with 3 to 5V, uses I^2C bus [33] interface and has a MEMS accelerometer and a MEMS gyro.

Figure 36: MEMS 6-DOF Gyroscope

.

Gyrometer/Gyroscope

A gyrometer [41] is a device helpful to measure change in orientation with reference to the angular velocity [5]. The practical model was based on the earth's Coriolis Effect. The rotation of the Earth about its axis causes substantial effects of force, pressure on the surface resulting in movement of storms and sea currents. It is caused due to the angular speed generated from the earth's rotation. It also can be described as an induced momentum [27]. The practical model for this demonstration is below, and this kind of device can be seen in many physics laboratories.

Figure 37: Gyroscope

The apparatus just needs to rotate. As it rotates, it induces the other rims to turn generating the angular momentum. Using MEMS technology, engineers design the whole system in a small IC. It uses the inverse piezoelectric effect. The change in the vibrations of the crystal caused due to an induced momentum are entirely recorded from the amount of electricity it produced [26].

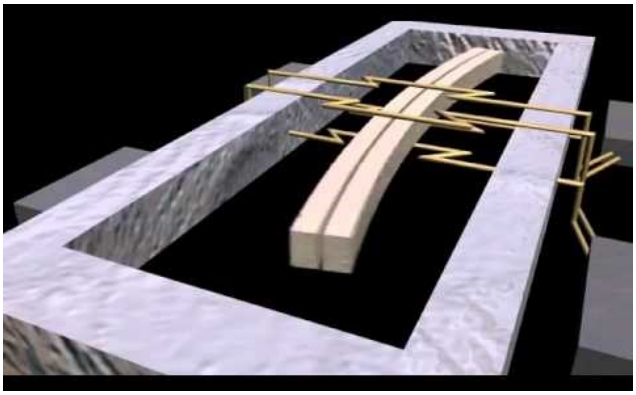


Figure 38: MEMS Gyroscope

The design contains a quartz crystal in the centre and two electrodes as the primary component. The drawback of a gyroscope is that it cannot determine gravity [5] [28].

Even though these IMU sensors have advantages and disadvantages, they can be made to complement each other when computing using all three sensors in an algorithm. These types of algorithms are technically termed as sensor fusion algorithms [28].

Unity–Arduino GPIO Test

The documentation of these tests explains progress was made through this research. The first Arduino test project was to check whether the programming boards work or not under the supervision of the researcher's guide, Jim Woods.

His Arduino Uno kit was used for initial learning [31]. IDE 1.6.5 was used for pushing the program into the board [33]. It is available on the internet as a free download and all its contents, libraries and examples are made free for innovators on their official website [33]. So it is commonly called an open source platform. As said earlier, the first test is to check the programming board is functional. This can be done by uploading a simple program that flashes the LED on the Arduino. The program is available as a primary example file in the Arduino library under the name "Blink". Before loading the program, two necessary steps are needed:

1. To specify the baud rate.
2. To specify at which communication port (COM) the board is talking.

After doing that, the example program is loaded or the following coding used:

Basic Arduino test program [33]**:**

```
void setup() {
pinMode(13, OUTPUT);
}
void loop() {
digitalWrite(13, HIGH);
delay(500);
digitalWrite(13, LOW);    voltage LOW
delay(1500);           }
```

After loading the program to the connected board, the LED should blink according to the delay indicated in the coding. The delay can be changed to understanding further the interface. If the LED is not blinking, then the board might be faulty, or the initial steps need to be checked again [32].

The Unity–Arduino test game

After this basic explanation from the supervisor, the idea was to create a Unity test game over the serial inputs from Arduino Uno. The primary challenge was to

research the method that would enable the Arduino to control the game made in Unity via a serial connection [27]. Two buttons and one potentiometer have been used. Coding is in Appendix 3 in The Unity–Arduino Test Game.

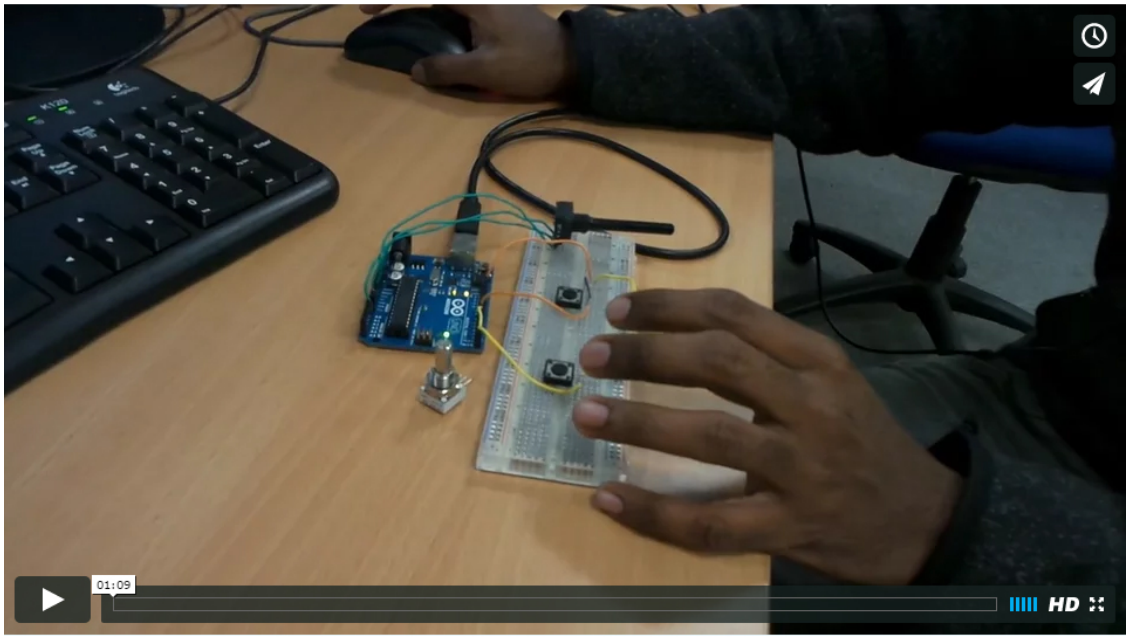


Figure 39: Test Game 1

The relationship is very simple: one terminal of a button is connected to a digital pin and the other end to the ground. For the potentiometer, three connections are made: here, it is powered up using the 5V supply from the Arduino, the signal line connects the analog pin, and the remaining terminal is grounded [33].

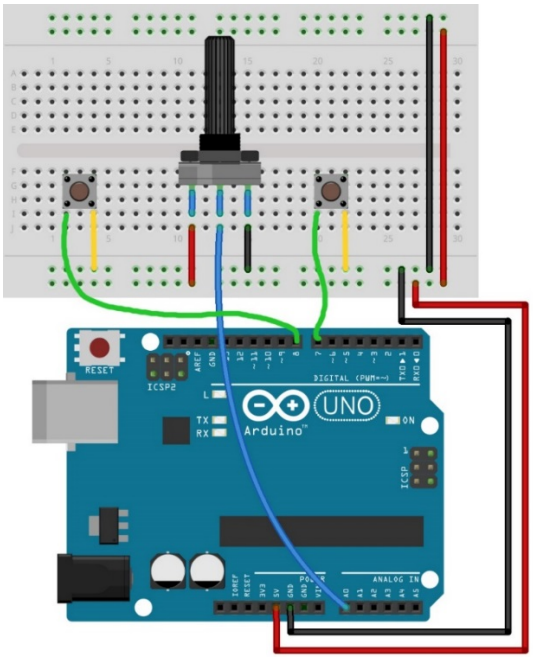


Figure 40: Arduino Connections – Test Game 1

The port is specified thus:

```
const int buttonPin02 = 8;
```

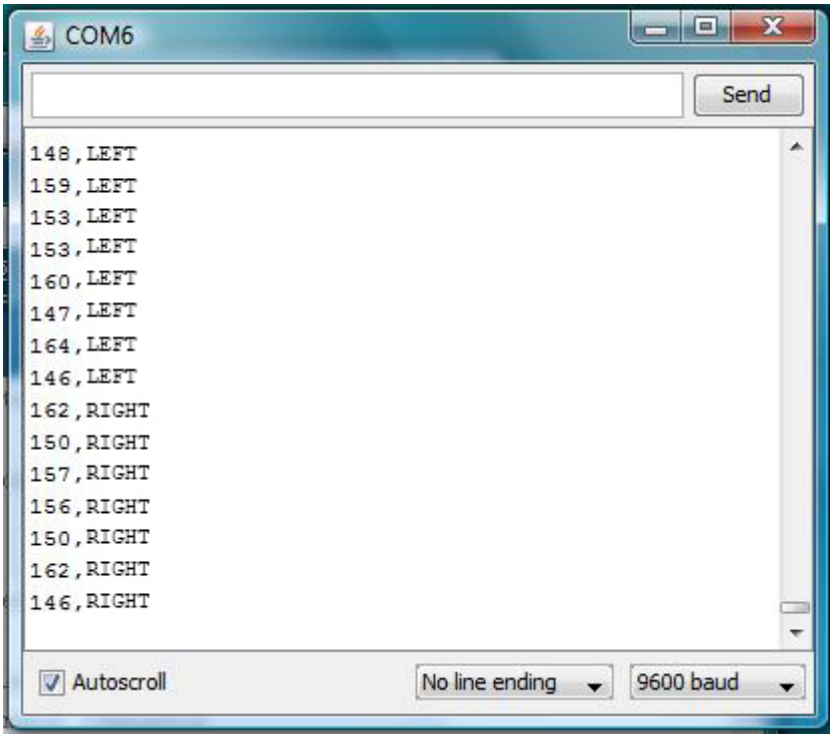
The baud rate is set to 9600. A program is run receiving the signals from the port and this is mapped within a range:

```
Serial.print(map (analogRead(0),0,1023,0,359));
```

Then it is printed to the serial data along with the serial button:

```
Serial.print(",");
```

After compiling and uploading the program to the board, the readings can be seen from the serial monitor, and the text interactively altered in the event of turning the knob and by pressing the button.



On the Unity side, the stage is done quite easily with an animating character called “Goober.” Initially, this game was made for the keyboard. Later, the program was altered to support the Arduino-based game controller.

Unity Code for Arduino controls [34]:

```
using UnityEngine;
using System.Collections;
using System.IO.Ports;
public class GooberSerial : MonoBehaviour {
	SerialPort stream = new SerialPort("COM5", 9600);
//Set the port (com5) and the baud rate (9600, is standard on most devices)
	Vector3 rot;
	Vector3 offset;
	void Start () {
		stream.Open(); //Open the Serial Stream.
	}
```

```
// Update is called once per frame
        void Update () {
string value = stream.ReadLine(); //Read the information
string[] vec3 = value.Split(','); //Arduino script returns a 3 part value (IE: 12,30,18)
rot = new Vector3(0,float.Parse(vec3[0]),0);
                        //Read the information and put it in a vector3
transform.rotation = Quaternion.Slerp(transform.rotation,
                        Quaternion.Euler(0,rot.y,0),Time.deltaTime*3);
switch(vec3[1])
                {
                case "LEFT":
                        GetComponent<Animation> ().Play ("walk");
                        transform.Translate (0, 0, -0.2f);
                        break;
                case "RIGHT":
                        GetComponent<Animation> ().Play ("walk");
                        transform.Translate (0, 0, 0.6f);
                        break;
                case "IDLE":
                        GetComponent<Animation> ().Play ("idle");
                        break;
                }
                stream.BaseStream.Flush();
                }
}
```

Further, to make the game more impressive, the character was duplicated, so one character is controlled using the keyboard, and another character uses the Arduino-based controller [32]. Simply, one of the above scripts was applied to one character and another to the next character. It would be nice to add a ball to the scene so the characters can play stealing the ball from each other.

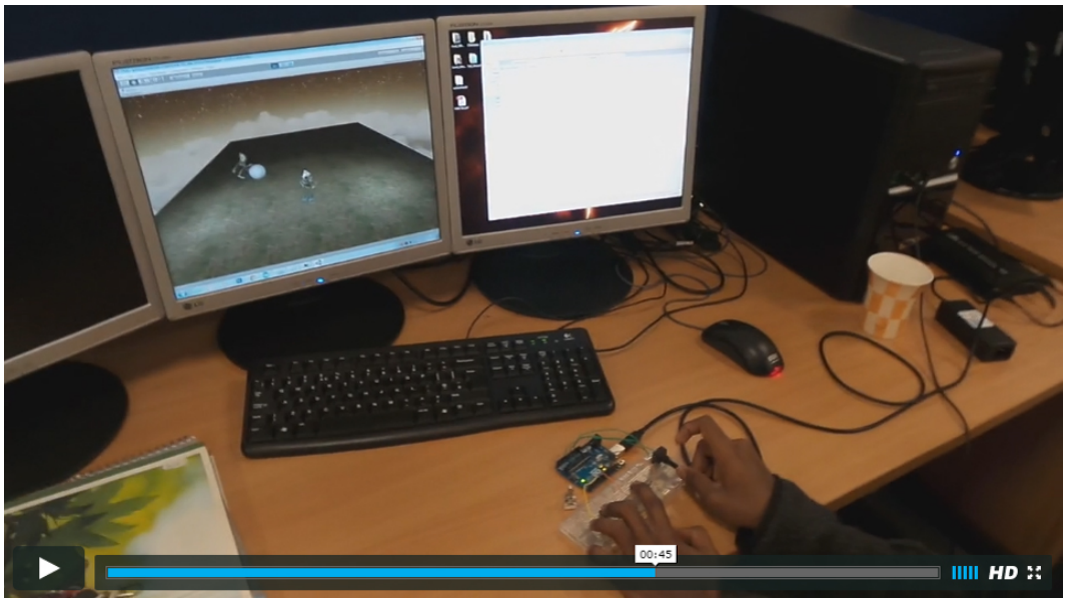


Figure 41: Arduino – Test Game 2

Along with a colleague, a documentation video was made and uploaded to the internet, at https://vimeo.com/142720649

The Unity–Arduino gyro test game

After the small success, the sensor-based inputs for a game were tested, using the same Arduino Uno board. This experiment was submitted to the researcher's supervisor, who was requested to lend his gyroscope to test it. The component received was quite a tricky one. It is not a direct gyroscope; it is a shield for the Arduino called "Tinker Kit" [33] which comes with various connectivity and sensors [33]. The Tinker Kit has its library with the pack or the library is also available on the internet for free. The sensors can be plugged in and the corresponding source code loaded from its example library to operate any sensors from the kit. The gyroscope it uses is a two-dimensional and direct one that connects to the analog port. It made the work simple here as the researcher already knew to interface the analog port because of the experience from using a potentiometer before [26].

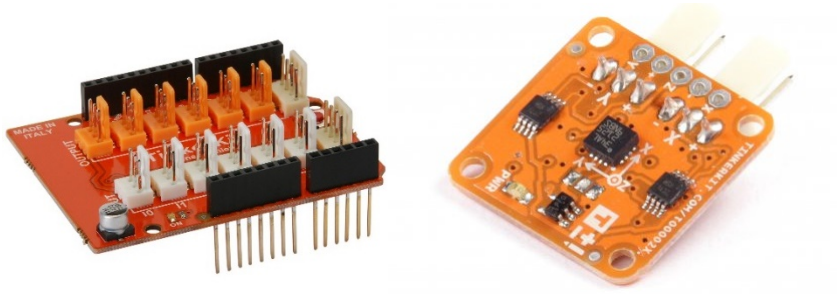
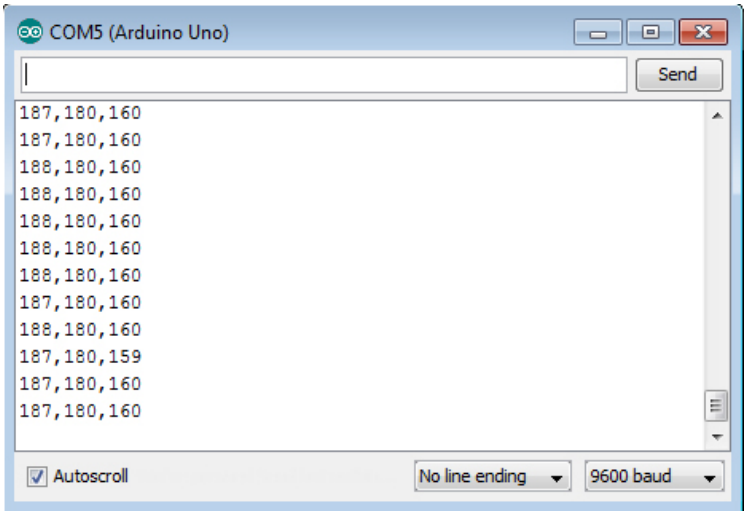


Figure 42: Tinker Kit Shield

The shield needs to be inserted on the top of the Arduino. The library for the Tinker Kit was be copied to the directory of the Arduino Library. When the connection was established, the gyroscope program was loaded from the library to the Arduino. The serial monitor shows the two-dimensional reading, and the reading should be changing as the sensor is moved. The source code from the library is below:

Specify the port and baud rate and print using gyro.read function [33]

```
Serial.print("X raw: ");
Serial.print(gyro.readX());       // print x-axis analog value
Serial.print("\trate: ");
Serial.print(gyro.readXAxisRate());    // print the x-axis anguar rate in the range of -/+1500 °/s
Serial.print("\t|\t Y raw: ");
Serial.print(gyro.readY());       // print x-axis analog value
Serial.print("\trate: ");
Serial.println(gyro.readYAxisRate());  // print the x-axis anguar rate in the range of -/+1500 °/s
```



On the Unity side [22], the gyro needed to be demonstrated in a game. As a demonstration, a pool board with one hole was created. To make the game interesting,a maze was created around the hole with mines to add a force to whatever collided with it. The entire game object rotated when the gyroscope was moved. A small ball was kept on the scene that has gravity. The motive of the game is to drive the ball into the hole, passing through the maze and mines. Coding is in Appendix 3 in The Unity–Arduino Gyro Test Game.

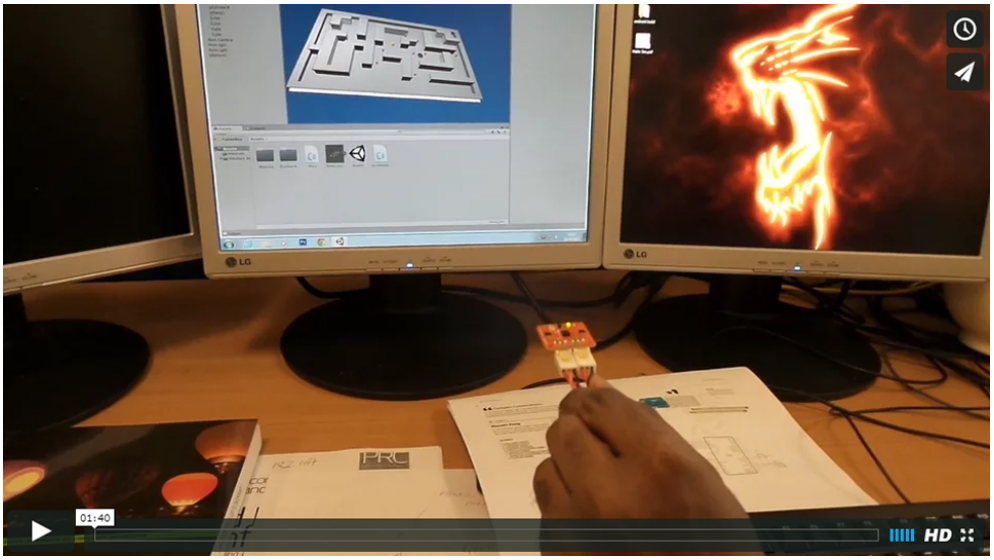


Figure 43: Arduino Test Game 2

The “serialRotation” script was attached to the game object which contains the pool board, maze, and mines. These mines alone were appended to a script called “force.”

The method to get the rotation on in unty using Euler method [22][34]:

```
string value = stream.ReadLine(); //Read the information
                string[] vec3 = value.Split(','); //My arduino script returns a 3 part value (IE: 12,30,18)
                if(vec3[0] != "" && vec3[1] != "" && vec3[2] != "") //Check if all values are recieved
                {
                        rot = new Vector3(float.Parse(vec3[0]),float.Parse(vec3[1]),float.Parse(vec3[2]));
                        //Read the information and put it in a vector3
                        transform.rotation = Quaternion.Slerp(transform.rotation,
                                        Quaternion.Euler(-(rot.x - temp0),rot.y -temp1,rot.z - temp2),
                                        Time.deltaTime*3);
                        //Take the vector3 and apply it to the object this script is applied.
                        stream.BaseStream.Flush();
```

The calibration part was implemented later on to reset the game again to its original position. It allows the advantage of holding the sensor freely without worrying about the orientation. A key “k” was assigned to do that. The GUI function prints the sensor reading on the screen while gaming.

This video documentation is also available on the internet, at:

https://vimeo.com/142723498

The remaining coding is in Appendix 3.

The Unity and analog joystick

The joystick is a simple device which works using two potentiometers where both the potentiometers measure the varying resistance as they are rotated [33]. The joystick that impressed this researcher was the two-axis Joystick Module X Y Arduino Dual Axis UK AVR PIC ATMega that comes with a push button as an additional event. This was chosen this because it can trigger five incidents in a single module. It uses the analog ports from the Arduino, so the device is precisely compatible with the working model.

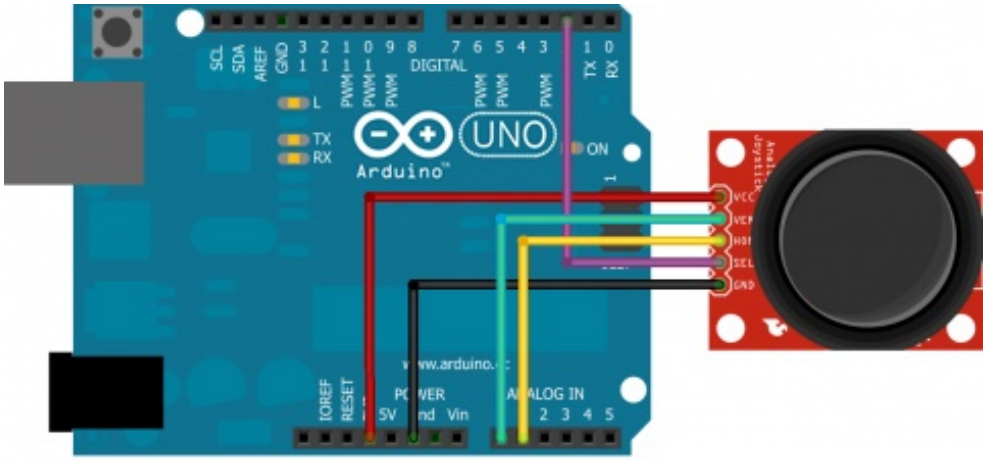


Figure 44: Joystick

The joystick test is easy to connect: the VCC to 5V and Gnd to GND; the analog ports A0 and A1 are connected to VRx and VRy, respectively; SCL to digital port 2 for triggering the button press event. The joystick test code is provided in the appendices.

The Unity coding is done by reading the string from the array and implementing a switch case statement accordingly. This was tested by applying the translation reacting to the printed input. The statement is in the appendices.

The commented lines are the alternate method. Both approaches were tried.

After completing the coding, I a 3D game object was created in Unity [22] and the script attached to it. It drove and moved the game object according to the

joystick signal. After accomplishing the full prototype, attention turned to building the game controller.

Coding is in Appendix 3 in Unity and Joystick.

Interface

Note: Everything documented is according to the chronology.

I^2C interface

I^2C devices [33] are compatible devices which use the I^2C bus to communicate. The device should be compliant according to the specification mentioned under the documentation 9398 393 40011. The I^2C devices must have a serial address and should support I^2C protocols. (Ref: # 900405 Rev E, February 2013, ©2010 Honeywell International Inc.). All I^2C sensors for the Arduino operate with the maximum power supply of 5V [5]. The I^2C library needs to be downloaded to the Arduino directory and it is available free.

Testing I^2C magnetometer

From previous experience in connecting the Arduino, sensors and other components were purchased online, mostly from eBay and Amazon. The first purchase was magnetometer HMC5883L [33]. The connections are very simple where both SDA (Standard Data) and SCL (Standard Clock) are connected to the analog ports, respectively.

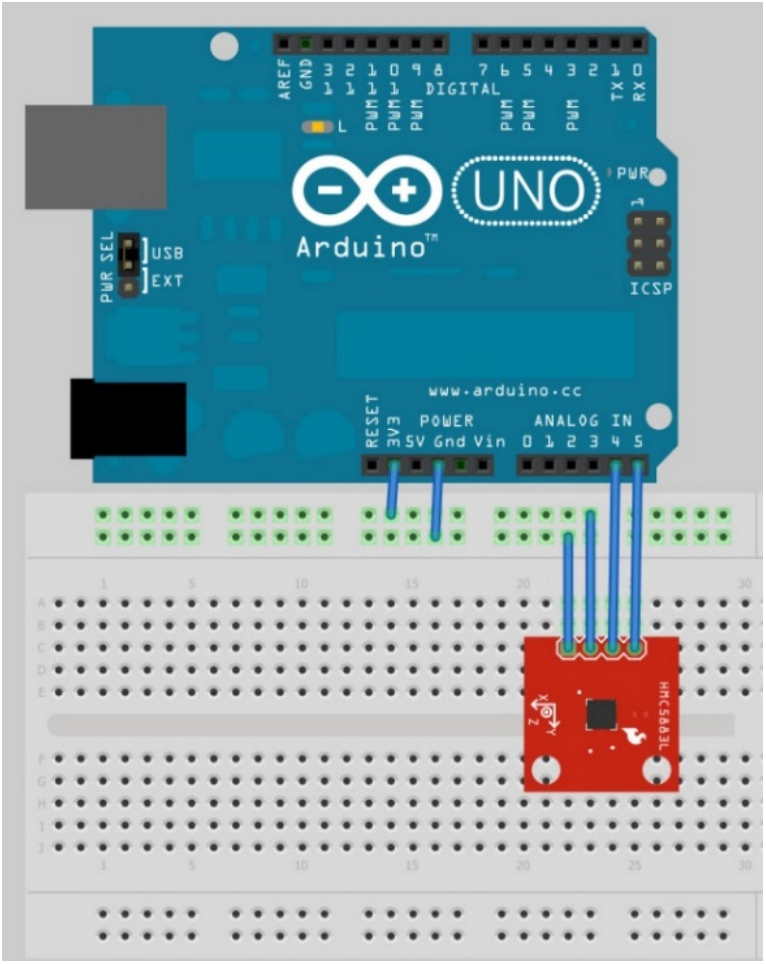
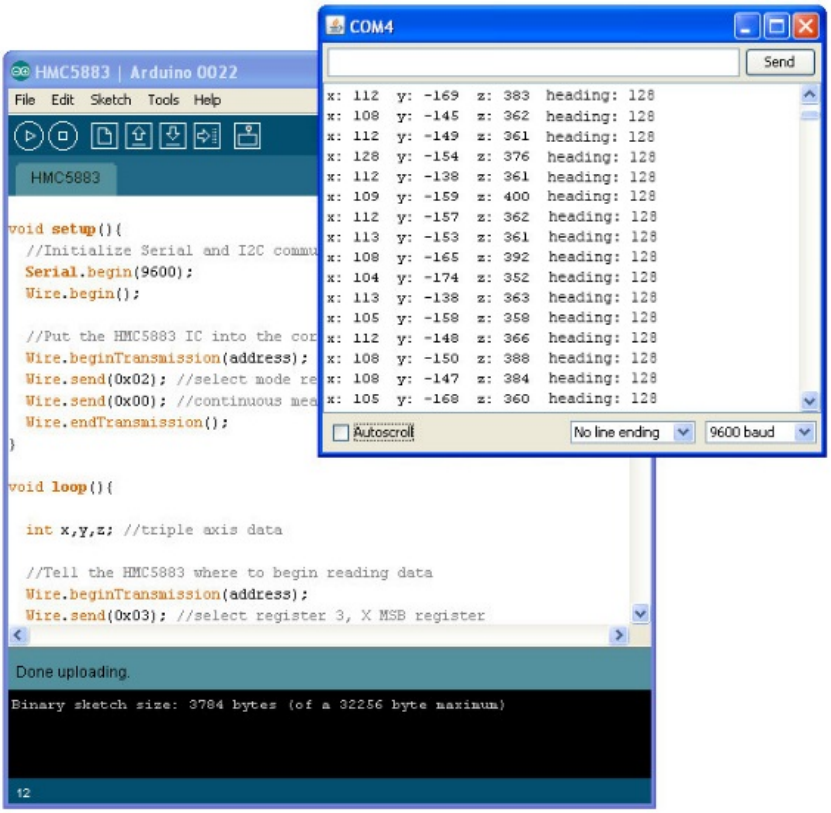


Figure 45: Magnetometer HMC5888L Connection

HMC5883l is a three-dimensional magnetometer [5]. It can be held in any position. The IC uses Honeywell's Anisotropic Magnetoresistive (AMR) technology (Ref: # 900405 Rev E, February 2013, ©2010 Honeywell International Inc.). It was the first sensor this researcher had used from the I^2C family. The sensor has a seven-bit serial address and connects to any device as a slave [32], so it needed a master to operate. In this case, the Arduino is the master for all I^2C-compliant sensors used here to achieve 3 degrees of freedom [27]. With this background information, the sensor was tested. The sensor usually comes unsoldered such that the decision is left to the user to choose how it shall be fitted. While soldering, the lines must not intercept. As this was this researcher's first experience of soldering, it was not without difficulties. A magnifying glass is useful to check. After the connection was made, the I^2C library was browsed for

the HMC5883L [27]. The corresponding coding is in Appendix 3 in Testing I^2C Magnetometer.

The code is interesting; it has I2Cdev.h which are the standard library functions for the I^2C program. The code for this sensor used the HMC5883L.h [33] library function to get the raw data from the sensor. After adjusting the baud rate to 9600 and altering the codes, the code was uploaded to the Arduino.



Testing I^2C MPU 6050

Similarly, the MPU 6050 sensor was purchased, which provides 6DOF, and is the best available IMU sensor on the market. It also belongs to the same I^2C family. The 6 DOF comprises the 3 degrees of the MEMS accelerometer and the 3 degrees of the MEMS gyro [33]. It connects to a device as a slave through the I^2C bus, and each channel contains 16-bits analog-to-digital conversion. The connection is similar to the HMC5883L except the INT, which must be connected to digital port 2 or 7 from the Arduino. The LED flashes as soon as the connection was made.

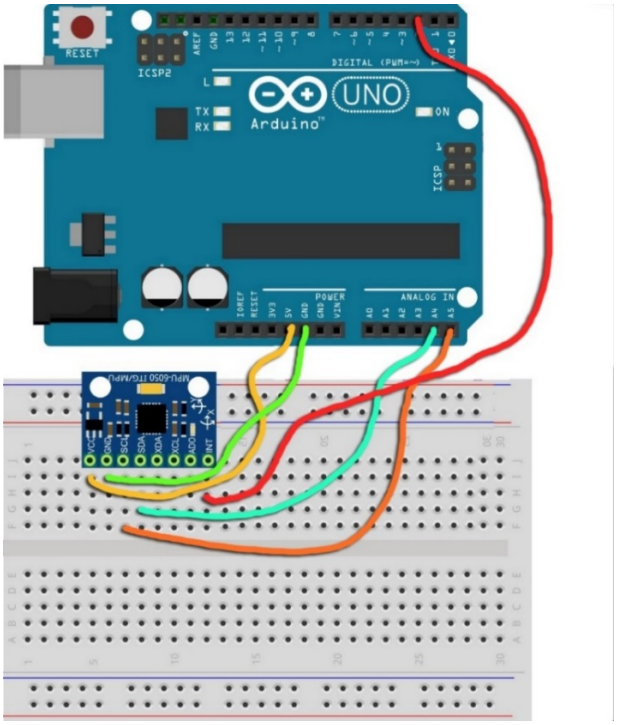
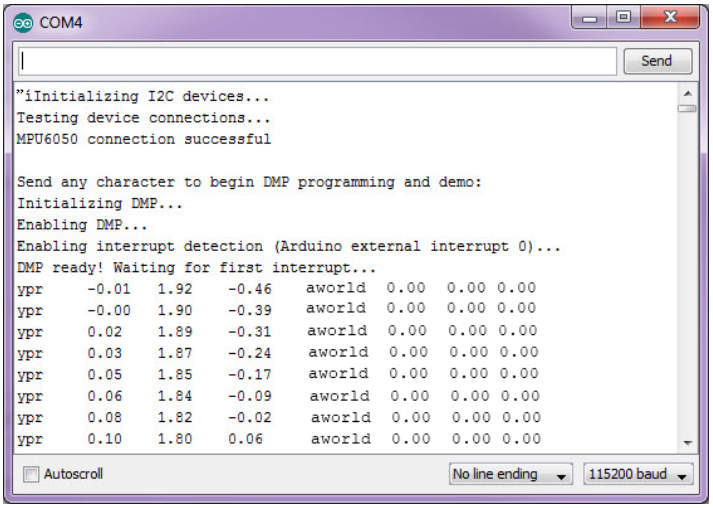


Figure 46: Gyro meter MPU6050 connection

The sample code to make it work also comes in the same I^2C library. On browsing the library, the test code for the MPU 6050 was found. The DMP system was tested. The example code provides the 6-axis motion fusion data in rotation matrix, quaternion, Euler angle, or raw data format [27]. The required declaration of the standard library function choose is uncommented, to run the code. Coding is in Appendix 3 in Testing I^2C MPU 6050.

The accelerometer reading was not convincing as it displayed four-digit readings on the serial monitor. The acceleration should have shown zero when there is no acceleration. On altering the coding, the serial monitor was made to display just the difference between the readings. Thus, the output was adjusted according to specification.



Now, the challenge was to connect the two sensors to the Arduino. Initially, the researcher believed it was really easy to do without proper understanding of the I^2C sensors. On the first attempt, it was anticipated the connections would be similar to the previous projects, but this was entirely wrong. All I^2C sensors connect the same SDA, SCL port of the Arduino. Researching [33] indicated that even though all sensors connect to the same pin, the Arduino determines the sensor by its clock. So it is possible to connect more than one sensor to the Arduino to the same serial bus. The following block diagram explains how the connection has to be made.

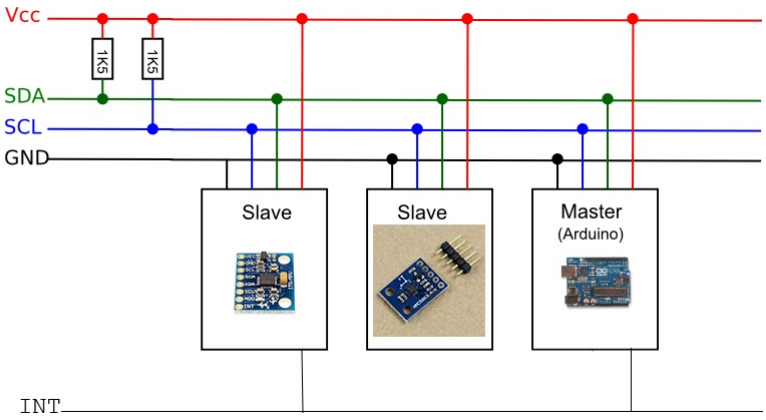
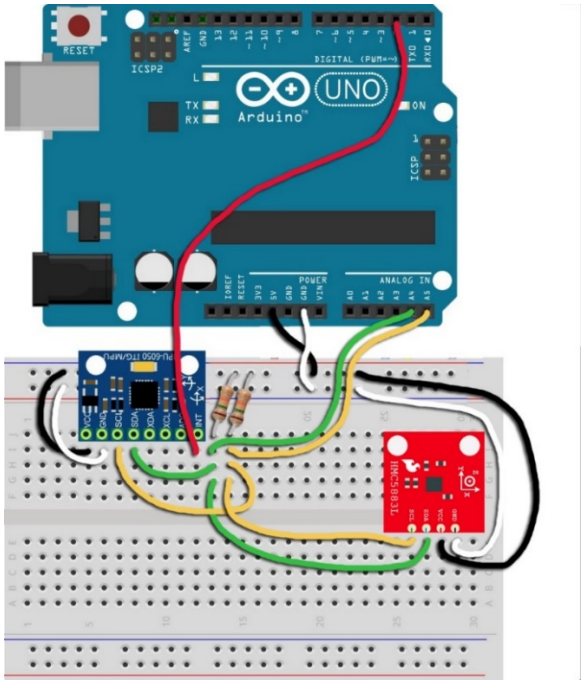


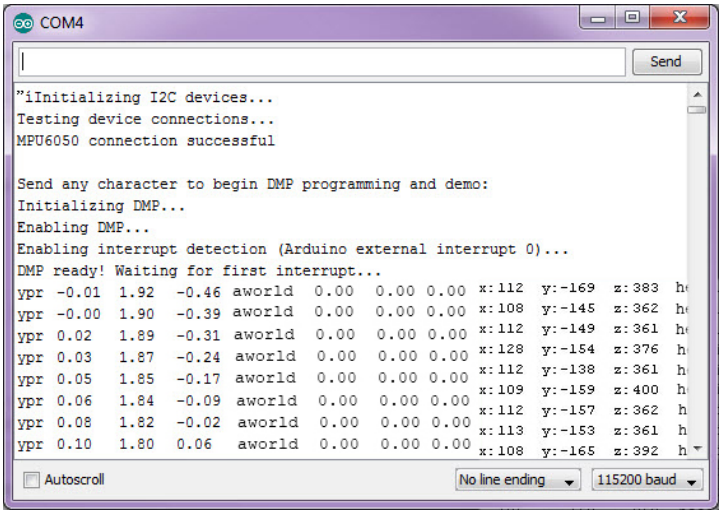
Figure 47: I^2C Architecture

Here, two resistors are embedded: one in between VCC and SDA and another between VCC and SCL. The sensors are slave-driven but the master is the Arduino Uno [31]. A breadboard was used for the prototype.

Testing with multiple I²C sensors



Connecting the resistor is not difficult because it is pole independent. Port 2 was connected to the INT from the MPU 6050 [33]. Analog ports A4 and A5 were connected to the SDA and SCL, respectively [32].



Wireless test

In programming, the two codings were combined to achieve 9DOF [26], which includes the 3 degrees of the magnetometer as well. After compiling and uploading the codes, in the serial monitor, nine readings were found which

altered according to the movement of the breadboard. After achieving the 9DOF, focus moved to the controller wireless.

Solutions found during the research were such as the XBee, Wifi, Bluetooth and Infrared [33]. As said earlier, the idea was to make a mobile-oriented VR game, so a decision was made believing the researcher's instinct that Bluetooth technology is more suitable for this type of project. Initially, the idea was to design a Bluetooth but, later on, the Bluetooth integrated module HC 06 was found, which is very easy to handle and is available in the market.

The Bluetooth module has its firmware compatible with that of the Arduino [32]. Hence, there is no need for writing the codes to the Bluetooth module. The HC 06 is able to connect a device only in slave mode.

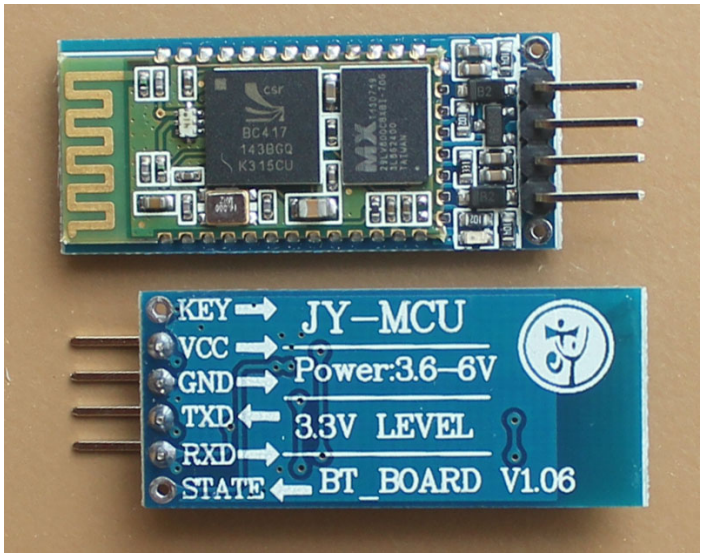


Figure 48: Bluetooth Module HC 06

The data transmissions are done through the data lines TX and RX from the Arduino. These were linked with the 0^{th} and 1^{st} digital port of the Arduino. The Bluetooth module also receives power and Ground from the Arduino through 5V/3.3V and GND, respectively.

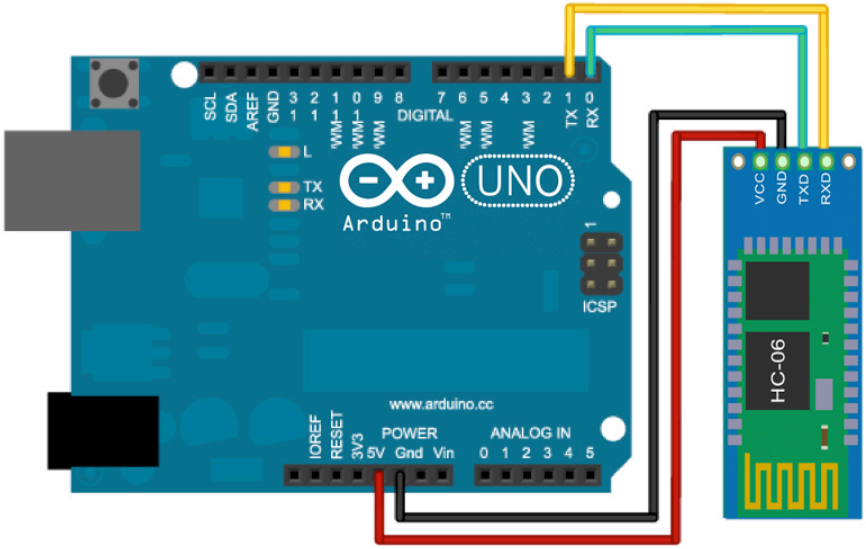
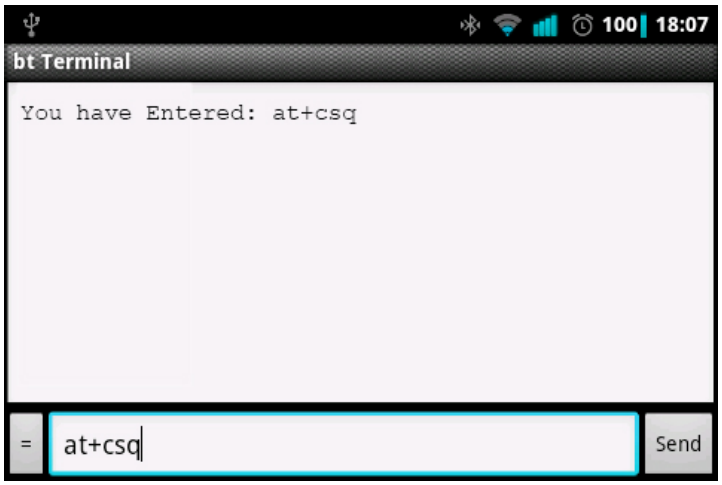


Figure 49: HC 06 Connections

The data connections were made vice versa. So the TX of the Arduino connects to the RX of the Bluetooth module, and similarly, the RX connects the TX of the HC 06. When all the connections were made, the LED started to blink, which indicated the module was ready to pair with Bluetooth-enabled devices such as mobile phones and laptops [2]. Scanning discovered the module under the name HC 06. If the pairing needs a password, “1234” is the default password. On the Android platform, an application called “Bluetooth Terminal” is used most commonly for this purpose.

There are no sample libraries for Bluetooth. Any code can be used to check the Bluetooth; it just needs to show the same readings wireless after it is paired. However, the researcher did a simple coding to testing the Bluetooth. It is just to echo the input string [33].

```
String message; //string that stores the incoming message
void setup()
{
pinMode(13,OUTPUT);
Serial.begin(9600); //set baud rate
}
void loop()
{
 while(Serial.available())
 {//while there is data available on the serial monitor
   message+=char(Serial.read());//store string from serial command
 }
 if(!Serial.available())
 {
   if(message!="")
   {//if data is available
     Serial.print ("You have Entered:  ");
     Serial.println(message); //show the data
     message=""; //clear the data
   }
 }
 delay(5000); //delay
}
```



The program is loaded before the pairing is made. The Arduino is connected with 5V power supply. When the device starts to talk, the LED stops blinking and flashes continuously which notifies that the connection was made. On typing anything and then Enter, the text shold echo back without the cable to interface.

After testing the Bluetooth module individually, all components were combined to form the circuit. Thus, the first prototype motion controller was made.

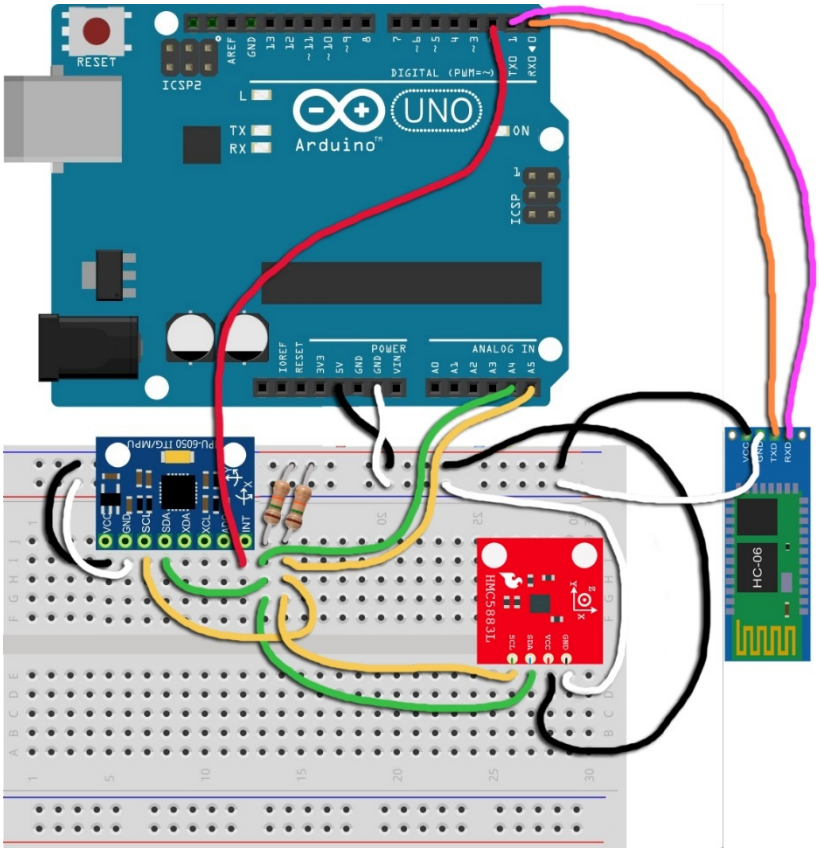


Figure 50: Motion Controller Trial Connection

On powering up the circuit with the 5V battery, the Arduino starts the LMC5883L, MPU6050 and HC 06. Now it is important to check at which port the Bluetooth is connected to the computer; port settings in the Arduino IDE were checked accordingly. The corresponding COM port was specified. The signals from the prototype now showed in the serial monitor. This is documented as a video here: https://vimeo.com/145580836

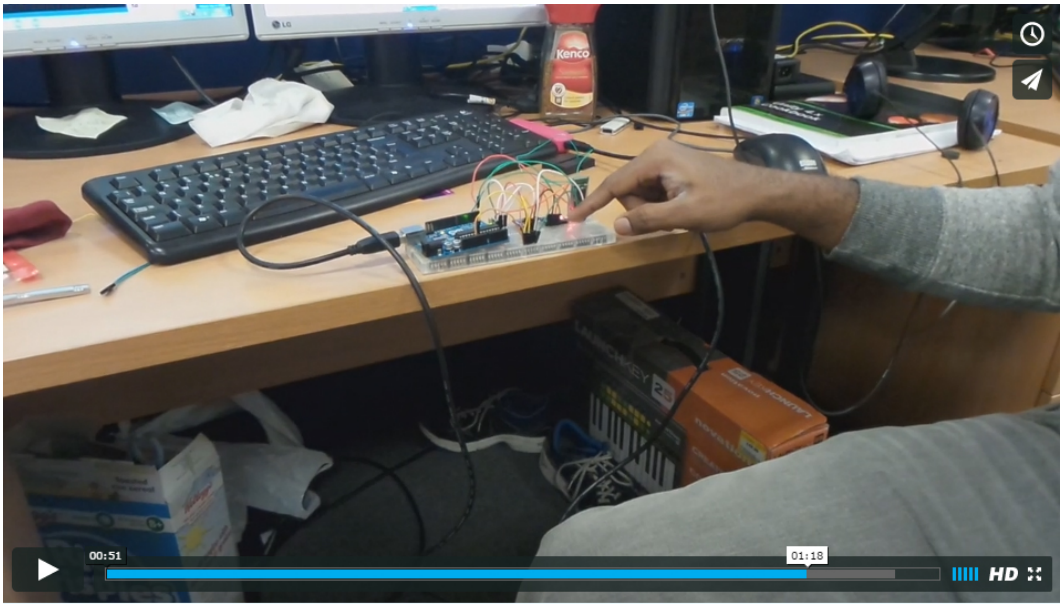


Figure 51: Motion Controller Trial Test

Initially, a 5V battery was used to power up the circuit. Then a Lithium-ion battery was considered for its size; then finally, idea came to use the power bank which used for charging mobile phones. The first power bank tested did not power up the circuit because it has additional features that are switched off when there is an inconsistency in the power. So a lower model was needed without this feature, so that powering up the controller does not cause a break. Also, the power bank can be used as the holder for the controller. Later, a new power bank was used with an on/off switch added to it.

Figure 52: 5V Power Bank to power up Arduino

Prototype and Implementation

Design trial

After testing the interface, to visualize how the final controller would look, a test mock-up was made, with work towards forming a circuit board base on it. While visualizing, the PCB (Printed Circuit Board) mounting [32], components and other peripherals that are available in the market for accomplishing the task needed consideration, and also the model should be be a solderless plug-and-play type.

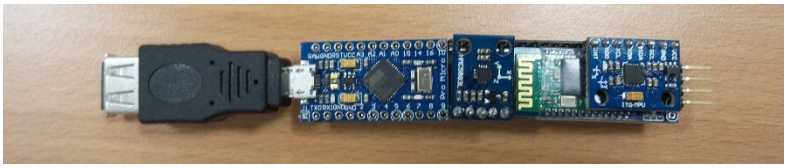


Figure 53: Design Trial

On mocking up, minimising space wastage was a consideration. The mock–up also showed that the controller could not be fulfilled unless a button was added to it, as that it would be useful to trigger at least an event. After browsing the link randomly [33], an idea formed to add a joystick to it. This could be placed it over and utilize the space above the USB adapter.

Compatibility test

Before starting implementation, the same prototype was tested changing the programming board to Arduino Leonardo because the components ordered were based on ATmega32u4 [33], as mentioned earlier under the topic “Motion Sensors”. The initial idea that everything would be similar was a mistake. While trying the same circuit with Leonardo [33], it became obvious that each board is different, needing research to solve this issue. On studying and comparing the connections, the circuit was adjusted accordingly. The instinct was that the connection would be the same, but the ports would be different in the Arduino.

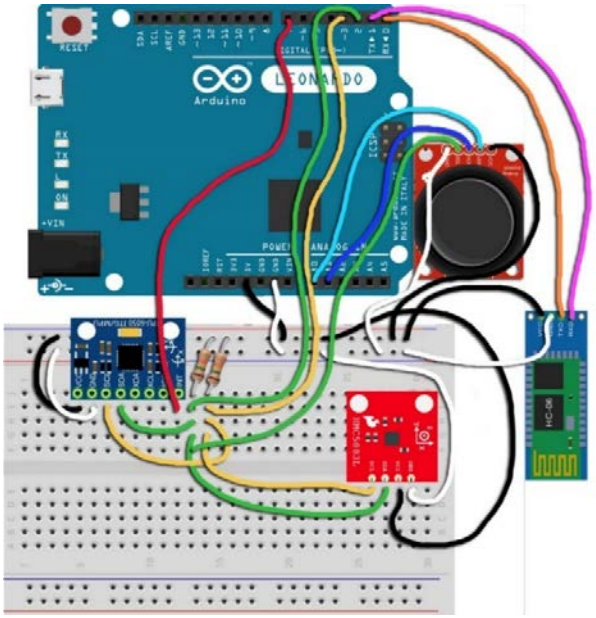


Figure 54: Compatibility test

The following connections were made:

1. Tx to Rx
2. Rx to Tx
3. SDA to SDA (which is digital port 2 in Leonardo)
4. SCL to SCL (which is digital port 3 in Leonardo)
5. INT to digital port 7
6. A0 and A1 were connected to VRx and VRy of the joystick
7. GND to Gnd
8. 5V to VCC

Even though all components were working, the Bluetooth failed to transmit data whenever the HC 06 was connected to the Bluetooth-enabled device. Pairing it with the devices worked, but the signal transfer did not. This proved difficult. Finally, attempting to power up using the Vin port resolved this issue. It was because of the UART at the port. Studied on the internet [33] showed how each programming board has a different architecture. For the Leonardo architecture, the design has only one UART. Whereas in the Uno, the design allows 2 UARTs. While using the Uno, either of the two UARTs or both UARTs can be used, according to the connections. So it can talk using Bluetooth while being

connected or powered through its USB port. But in the case of the Arduino Leonardo, it is tricky because this has only one port. To power the Leonardo using the USB port, the Bluetooth cannot be used because the one and only UART it has, has now been blocked. This can be resolved in two ways:

1. By powering up using the Vin port
2. By looking for an option to achieve this in the coding

Despite concerns, the second option was chosen, and later, a problem was discovered at testing, after the implementation and soldering were done. On looking for the coding option, a relevant example was found on YouTube [42] where the uploader left a piece of coding in his description. He used “Serial1.available()” instead of “Serial.available()” which gave a clue to try “Serial1.available()” instead in the coding. This solved the issues. In addition to this, changing “Serial1.available()” in the coding became apparent for using Bluetooth [32]. So, to connect Leonardo by cable, the codes do not need changing. All Leonardo coding should be following a library called “wire.h”.

On testing with the Arduino Pro Micro, the researcher was 100% sure that it would work because the same functionality in the Arduino Leonardo is stripped down in the Pro Micro, which uses the same ATmega32u4 [31] technology. The connections are the same as in the incompatibility test using the Arduino Leonardo.

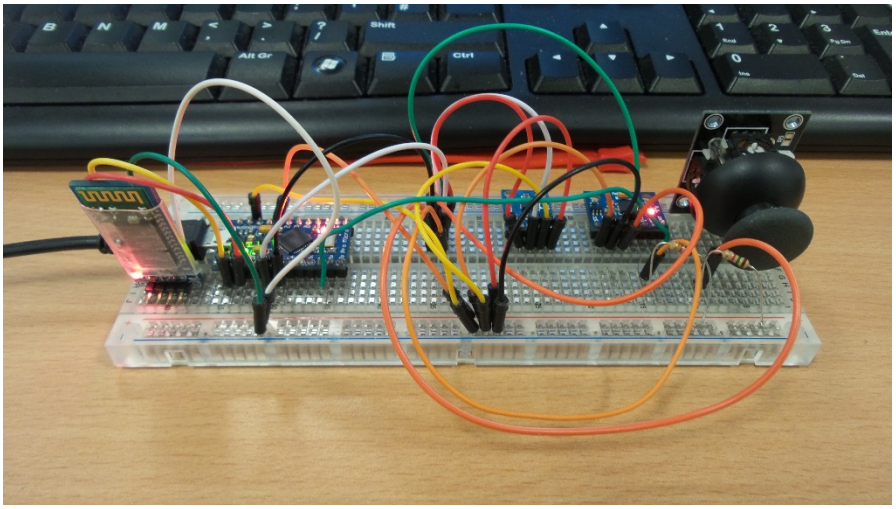


Figure 55: Final Prototype

The tests presented in this chapter guided the design of the motion controller and also have planted the idea of taking this concept further.

PCB Implementation

This discusses the implementation of the [2] motion controller and VR goggle. This covers:

- Designing and printing the PCB board
- Mounting and soldering the IC holders
- Implementation
- Testing and understanding
- Drawbacks, errors and measures

Designing and printing the PCB board (motion controller)

The first and foremost plan was to design in such a way that it should be held in the hand. The idea was to design for a space which is just double the power bank in length and breadth, for compatibility with holding [26]. Since the power bank fills half the space, there is just the remaining half to express these ideas. The idea was to design a wireless board, and not to solder any sensors permanently to the board because then the sensors can be replaced easily with new ones whenever there are issues and faults.

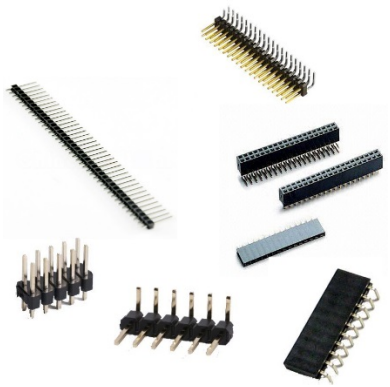
Mounting and soldering the IC

Researching indicated two possible types of PCB mounting, namely in [27]:

1. Pin
2. Sockets

A pin can be connected to a socket, and a socket can be connected to a pin, but it cannot connect to its own type. The pins are commonly referred to as male, and the sockets as female. One end of each has lead to solder such that it can be mounted on the PCB, which is why these are called PCB mountings. Based on these two basic classifications the mountings are classified as follows:

1. Straight single header pins
2. Straight single header sockets
3. Right-angle single header pins
4. Right-angle single header sockets
5. Straight double header pins
6. Straight double header sockets
7. Right-angle double header pins and
8. Right-angle doubles header sockets



Cables/jumpers were also used as a PCB mounting. Most of the cables also come with header sockets and pins. Either a connection can be made to another header or a solder can be made directly to the board. Depending upon the header connection the cables can be classified as:

1. Male-to-male headers
2. Female-to-female headers and
3. Male–to-female headers

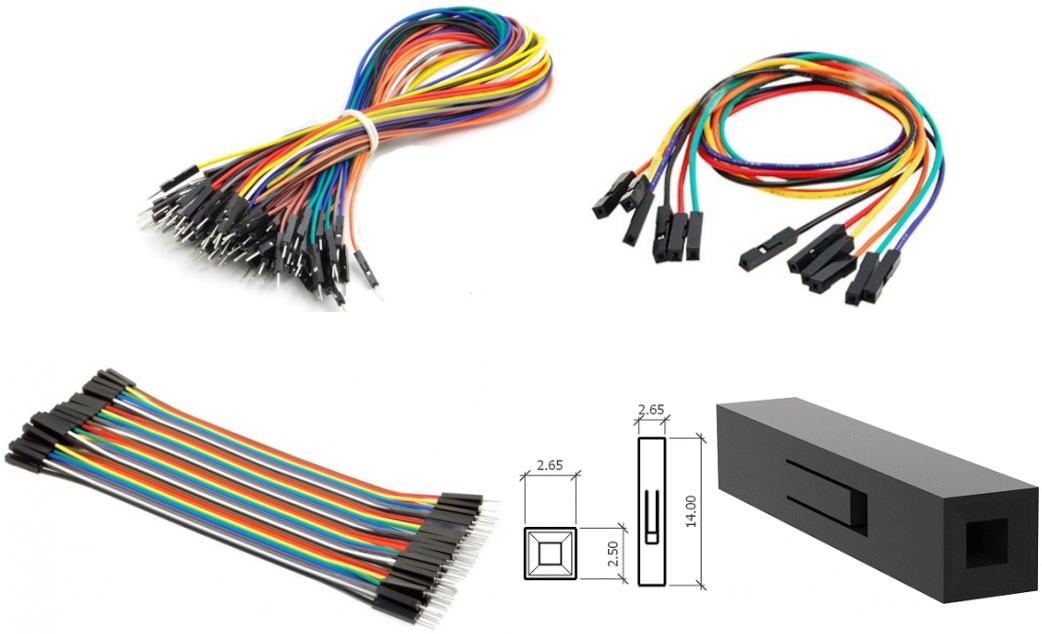
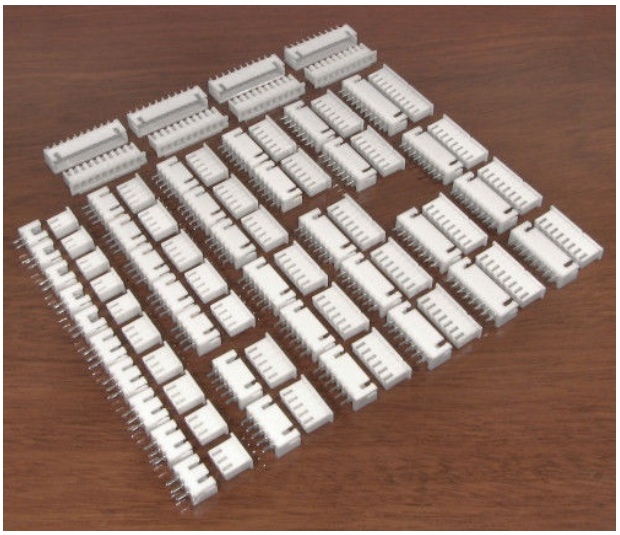


Figure 56: PCB Mountings

A few special type headers with a locking system which will give 100% assurance of connectivity and are shock proof were ordered. These are usually callled plugs. Sometimes plugs are a bit difficult to remove. A screwdriver pulls them out easily. All these items are available on the market but need to be purchased in packs because they are not available loose.



Apart from these mountings, there are special mountings to hold the ICs which come in many equivalent combinations like eight, twelve, twenty-four, forty, etc.

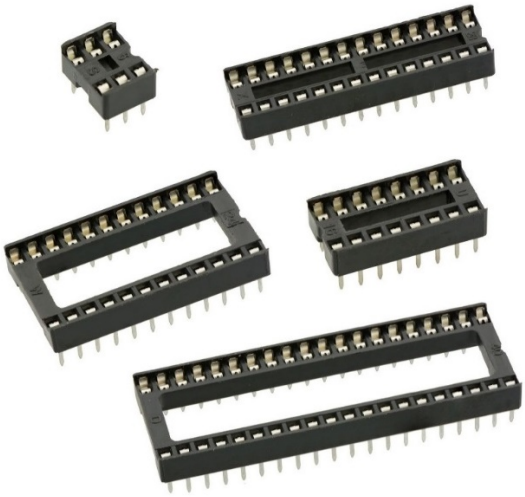
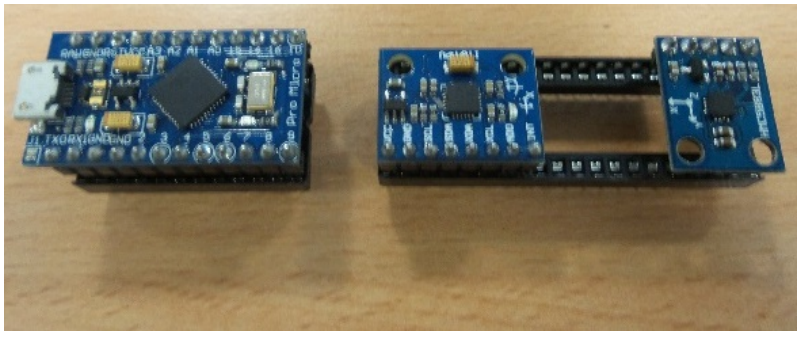


Figure 57: PCB Mountings 2

Special purpose components are the female socket for 5V power, Ethernet, USB 2.0 and USB 2.0 mini, etc. These also come under PCB mountings because they are mounted and soldered on the PCB.

Implementation (motion controller)

In this design, two IC holders were used. One is for connecting the Arduino board, and the other connects the sensors [32]. This used a four male pin header for connecting the Bluetooth and a five male pin header for connecting the joystick. The magnetometer was connected to the top left side of the 40-pin holder and the MPU6050 to the bottom right of the same holder. A 24-pin holder was used to hold the Arduino Pro Micro. The picture below shows how the sensors are held in the IC holders.



After the mock-up, a design was implemented in an application called PCB Wizard 3, which helps with drawing the circuit. Mountings and resistors can be drag-and-dropped from the PCB component gallery. The main window has tools to draw the circuit at the top and different types of views such as normal, real world, artwork, prototype etc. at the left-hand side. The application is simple enough to learn while designing.

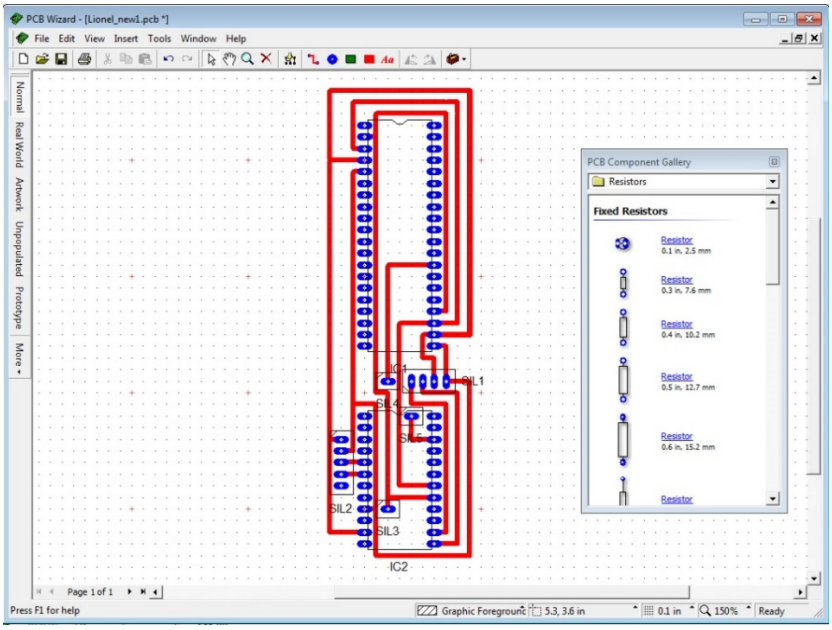


Figure 58: Motion Controller Circuit Diagram

After designing the circuit, the design was exported in the Gerber format, which is the most common format for printing PCBs. Exporting is easy, at Tools>CAD/CAM>Export Gerber. Then this researcher produced his first-ever PCB printing.

Figure 59: Motion Controller Circuit Printed Board

The next stage of implementation is soldering the mounting. On observing initially, this seemed an easy task, but later the need to have the skills for this work became clear. Before going for soldering, the holes needed drilling and holders and pins mounted as per the plan. Then the sensors and the Arduino were mounted on the holders and the functionality tested.

Testing and understanding (motion controller)

At the initial stage, on testing the circuit no reason was apparent why it was not working. Checking the connections indicated a mistake in connecting the resistors. The resistors were removed and the circuit adjusted manually. The plan was to add the resistor later, hoping the components would not get burned before adding those resistors bypassing the PCB.

Drawback, errors and measures (motion controller)

Although each and every connection was checked and adjustments made for the errors accordingly, there were still difficulties in fixing the circuit. Measuring the currents in each terminal revealed that it measured 0.69V across the circuit whereas it needed to be 5V throughout the circuit. Finally, the problem was located in the soldering on the Arduino side. The channel started to work after dusting off a bit of lead which was shorting the entire circuit. Issues mentioned earlier with the single UART were also resolved.

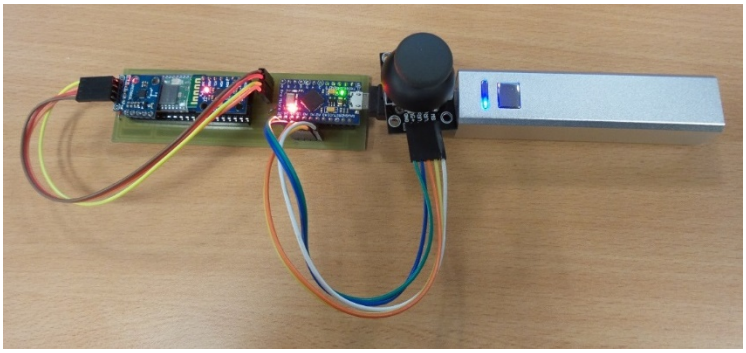


Figure 60: Motion Controller

In the final stages, a way to add 1.5K resistors [33] to the circuit was implemented by: drilling a 1 mm hole in the pathway of SCL and VCC between the terminals; then bypassing a 1.5K resistor in between them and soldering this; similarly, for the other 1K resistor connected the SDA to VCC. Small cables were used to connect port 7 with INT and SDC with the joystick socket header. This is how a flawless design was implemented.

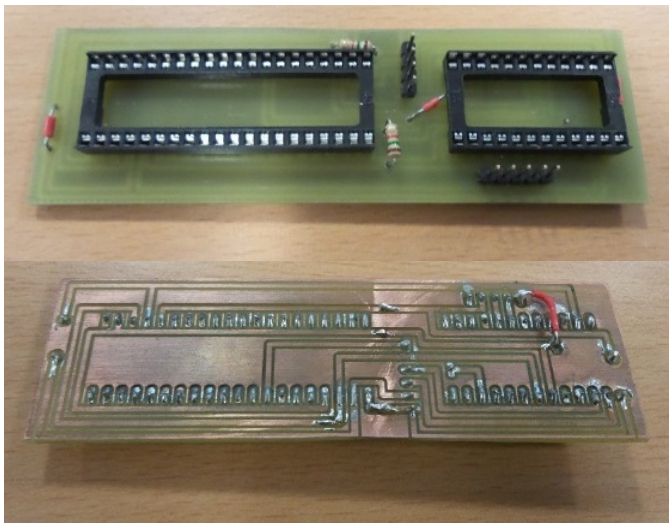


Figure 61: Motion Controller PCB Mountings

Jumper headers were used to connect the HC 06 Bluetooth module and the joystick. The video documentation is available at https://vimeo.com/151254967

The design was fabricated using the PCB printer Denford PCB Engraver, under supervision.

Implementation (voltage step-down module to 5V)

It is simple to develop a step-down from any voltage to 5V. It was anticipated that high voltage could burn the MEMS sensors and the Arduino. This module was used for the VR HMD where regulating 12V power from the LCD controller board was needed. The circuit contains a 5V 1A low dropout fixed positive voltage regulator and two 15uF100v capacitors.

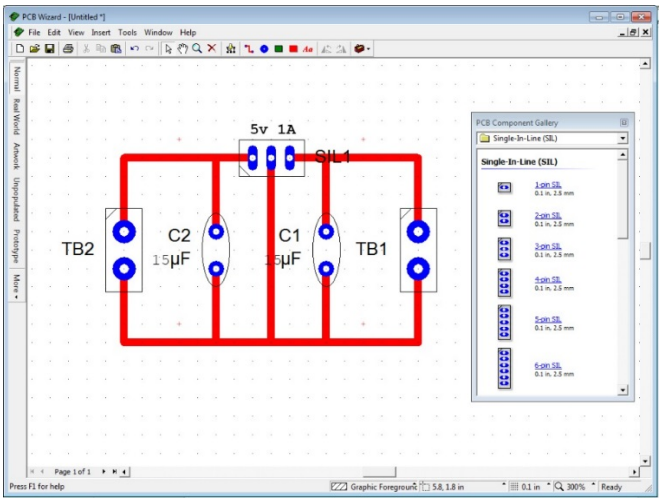


Figure 62: 5V Step-down Regulator Circuit

The voltage regulator was mounted at the top terminal and two capacitors, as mentioned, in the circuit diagram. The terminal TB1 now outputs 5V whenever 12V is sent from TB2.

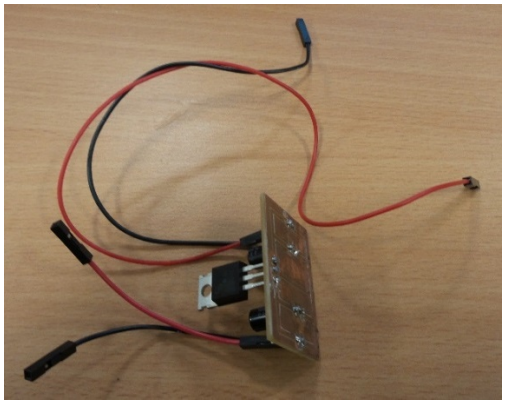


Figure 63: 5V Step-down Regulator

9DOF sensor

On looking for a 9DOF MARG sensor [11], an IMU sensor was purchased with a L3GD20H 3-axis gyro, an LSM303D 3-axis accelerometer, and 3-axis magnetometer embedded in a small chip. The MARG is also called a Pololu MiniIMU-9 v3.

This was easily tested because the sensors are I^2C type [33] and its library is quickly accessible. The connections are simple and familiar, and were discussed earlier in the Arduino I^2C test chapter. Plugging is as follows:

1. SCL and SDA to the analog lines 4 and 5 in UNO or digital lines 2 and 3 in Leonardo
2. Vin to 5V and GND to Gnd.

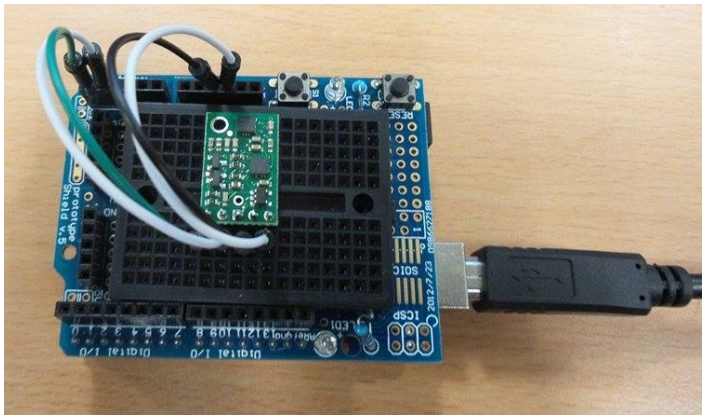


Figure 64: Connection MARG 9DOF Sensor

Once the connection is made, the following steps are taken: check the baud rate, load the library and run the program. The library has classes Compass, DCM, I^2C, Output, Vector and Matrix.

The links to the library are:

https://github.com/pololu/minimu-9-ahrs-arduino

https://github.com/pololu/lsm303-arduino

https://github.com/pololu/l3g-arduino

The MARG device was tested using the library above. Later on, Madgwick's AHRS algorithm was used to check and the result evaluated.

For testing, the output was adjusted according to requirements, using a function call to print the filtered quaternions to Unity. In Unity, a simulation was created, to move a drone using the serial data. As a result, movement with interactionwas achieved, and the orientation became aligned with the earth's magnetic north. Hence, the device performs with the right orientation while facing north.

3D printing

Initially, the plan was to 3D print the HMD, but after knowing the availability of VR gear in the market this idea was dropped. The intention was to fit the 7-inch display which comes with the LCD control board, but this ended with disappointment on purchasing many which were not compatible with the specification.

As as alternate idea, a partial 3D print was add to the HMD case purchased.

Figure 65: VR Gear

The display holder was chopped away and a new holder printed for the 7-inch screen and circuits.

The design was modelled and exported in a .stl format. The necessary parts were printed using a material called PLA (polylactic acid), which is a brittle.

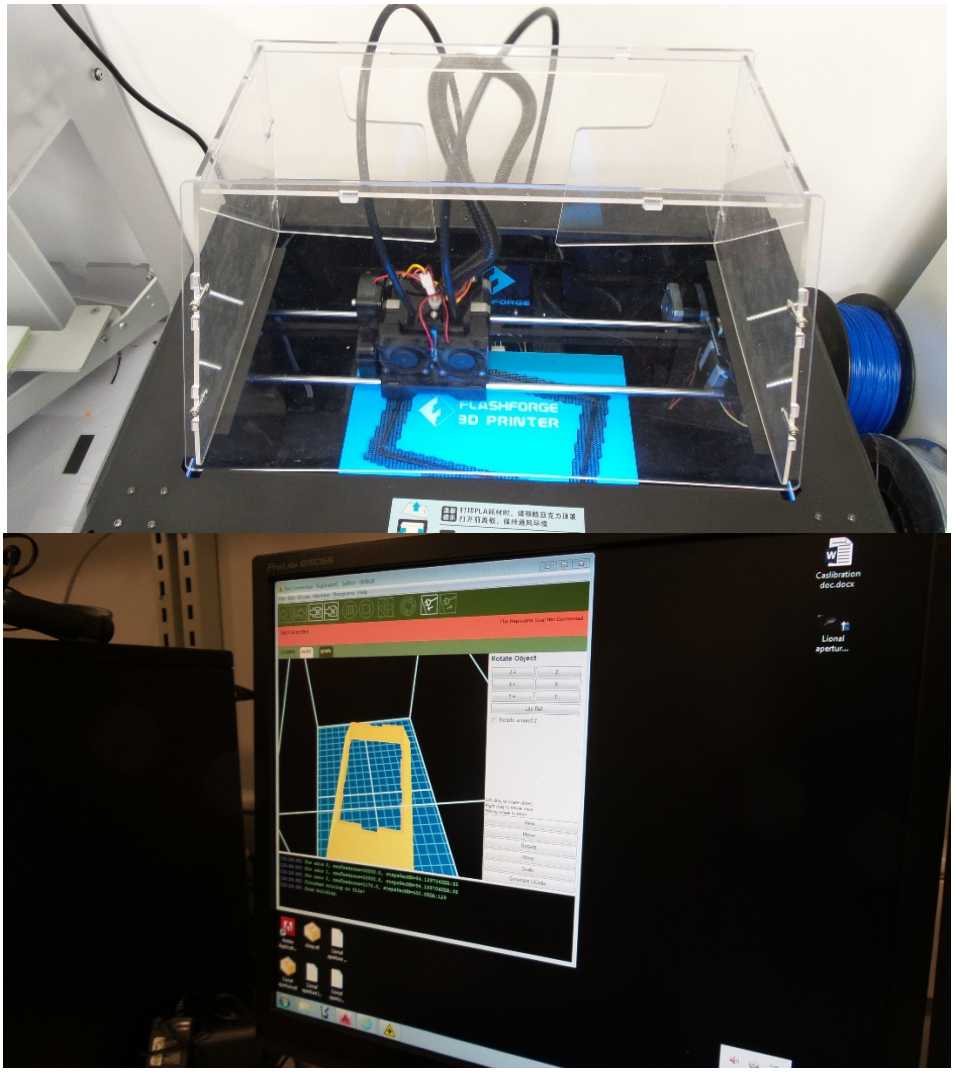


Figure 66: 3D Printing

3D printing is a time-consuming process which requires much patience. Thus, this is how this researcher learned to print models.

MHL interface

While developing the motion controller, the researcher believed that he could make a VR goggle in parallel. The intent initially was to demonstrate it with an Android phone [2]. The smartphone within the researcher's budget was purchased. At the time, he hadn't anticipated that the phone would become a significant investment in these studies. The choice was made with his aesthetic for photography, since Samsung S4 zoom was a hybrid between a phone and a digital camera. There are, he found, more attributes in the sensor. The decision to build VR for Android was made because most of the phones on the market are Android and the researcher's phone is one of the powerful mobile phones. As he started using the phone for the research, he discovered that it is perfect for the research because it satisfies all the key areas of his interest. The Samsung S4 Zoom is compatible[1] with:

[1] http://www.samsung.com/global/microsite/galaxycamera/s4zoom/specifications.html

Figure 67: Android Phone with MHL

1. Accelerometer
2. Gyroscope
3. Android KitKat
4. Bluetooth 4.0, IR and WiFi
5. OTG using MHL technology
6. 16-megapixel zoomable camera with augmented reality.

Later research showed that these specifications were not available on any other phone . Research showed that VR technology is available for Samsung phones which use the phone's in-built gyroscope [5] for specialized VR gaming experience. But the intention was to innovate a version of a VR goggle that can connect all mobile devices. In other words, a specialised VR device for gaming that tracks the movement using the mobile's processing power [4]. The desire was to achieve this VR goggle by adapting the same technology the researcher is familiar with and has learned in developing the sensor-based motion controller.

According to the gameplay architecture, the Android phone should speak with the VR Head Mounted Display (HMD) as well as the motion controller simultaneously. It seemed likely that video would struggle to stream wirelessly along with the interaction. It is always best to use cables, so here using HDMI between the VR HMD and the Android phone to stream the video to the VR HMD. The sensor's serial data should be connected to the Android phone to

trigger the HMD. The demand leads to search for a solution and current technology that suits the project. The choice was Mobile High-Definition Link (MHL)[2], which is used for playing mobile games on a big screen without any lag. It also streams high definition video and audio using HDMI technology.

The technology attempts to turn the mobile into a gaming console or controller. The first-ever product from MHL tried here was a Cable Adaptor Micro USB A MHL HDMI S3/S4, which is a 2-metre cable. While testing after purchasing, it was discovered that it is an active MHL cable primarily designed for the ease of charging the S4 phone while using the HDMI. So the USB end needed to be plugged into a power source all the time to operate the cable or else the cable would be dead and useless. It was a mistake to believe that the USB would support data transfer, as it is merely designed to power up the cable and does not have the two lines for data+ and data-.

Figure 68: MHL Technology

For the test, the phone was connected with the MHL cable [35] at the micro USB end, the other ends were connected to the power bank and the 7-inch LCD (HDMI). Meanwhile, the 7-inch LCD must also be powered up by a 12V battery to function. Now the phone can be mirrored.

[2] http://www.mhlconsortium.org/consumer.aspx

Figure 69: MHL Test

After testing, a MHL 5 was tried in one adapter for its On The Go (OTG) adaptation for the advantages of multiple ports. A USB 2.0 port seemed handy for implementing a serial communication between the Android and the VR goggle. This seemed achievable since the Arduino Leonardo Micro connects any device as a serial, such as a USB mouse and keyboard. The MHL [35] + OTG adapter comes with ports for SD cards, USB 2.0, Micro USB and HDMI. The interest was to try for maximum advantages of the architecture. On testing the architecture of the adapter, it was found that the design allows parallel interfacing. For instance, these adapters enable the phone to access the OTG and HDMI at the same time.

Added a mouse to the adapter, the mouse functioned on the mirrored screen.

Figure 70: MHL Adapters

On testing the other ports, it seemed a good idea if the adapter allows the phone to charge at the same time. So another type of adapter was investigated, which is twice as expensive as the one used so far.

Unfortunately, the attempt cost time and money as the description was not clear enough to ignore it. The new adapter solved the need, but it has a limitation such that it could not support parallel interfacing. A button is given to switch between the modes, so either HDMI or OTG is used at a particular time. In this architecture, the Micro USB is used for charging the adaptor only.

Attaching a USB hub to the USB port was also. It worked surprisingly well. It can connect 4 to 5 USB devices to the phone at the same time.

After trying out these interfaces, two models were proposed:

1. To add a data line to the original red MHL cable.
2. To pull out the raw wires to the circuit board on trying out the HDMI cables with Ethernet.

Due to the limited time for the research, this complexity was ignored; also the intention was for everyone to be able to do it with the available material on the market.

The design will become even simpler if it is possible to transmit serial data via Bluetooth. This would remove the need for an OTG adapter in the design. If using an OTG adapter, then the type of the HDMI cable in use matters.

There is a myth about HDMI cable. Some video tutorials advise that all HDMI cables are the same. This is true to an extent for general purpose usage, but this information is false and misguided the researcher totally. This wasn't figured out practically until trying out E324703 AWM Style cable. The cable gets the power from the display side. So the adapter need not be powered externally.

Figure 71: HDMI Cables

By this time, the design for the full working model of the wearable VR goggle were not finished, but as per the abstract design, the connections were simple:

1. Attached the display to the LCD controller board.
2. Power up the LCD controller board with 12V.
3. Step down the 12V to 5V from the LCD controller board and power up the Arduino Micro.
4. Add a button to the Arduino for the need for manual calibration.
5. The Arduino Micro connects the 9DOF sensor and transmits a signal via wireless.
6. The wireless receiver at the OTG end receives the serial data and instructs the application to move the camera according to the heading.

Filters and Estimators for Orientation

Absolute orientation

After assembling the sensors to form an inertial motion unit, the data need to be refined. The data from the all the three sensors move an object in a 3D space. The movements were interactively triggered but difficult to control because of the orientation. The headings or the precision should be calibrated according to the position of the IMU. This is known as the absolute orientation. To achieve absolute orientation is a core of the problem for this research discussion because there is no sensor to give the altitude easily. The Kalman filter (proposed in 1961) provides a solution to estimate or generate the data when there is uncertainty. After Kalman, the scholars Mahony and Madgwick took this challenge to the next level predicting the altitude by fusing the sensors and provided their innovative algorithms. This current work adopts one of the algorithms to get the maximum immersion to track the movement in the 3D space.

Kalman filter estimation literature

The conceptual idea of the Kalman filter [12] is to use a set of equations and continuous data inputs to estimate the actual value of the moving object in real time. It is an iterative mathematical process that filters the errors due to uncertainty such as noise and other factors.

The flow chart below explains the working principle of the Kalman filter. The “update estimate” generates the new iterative values each time by processing the algorithm.

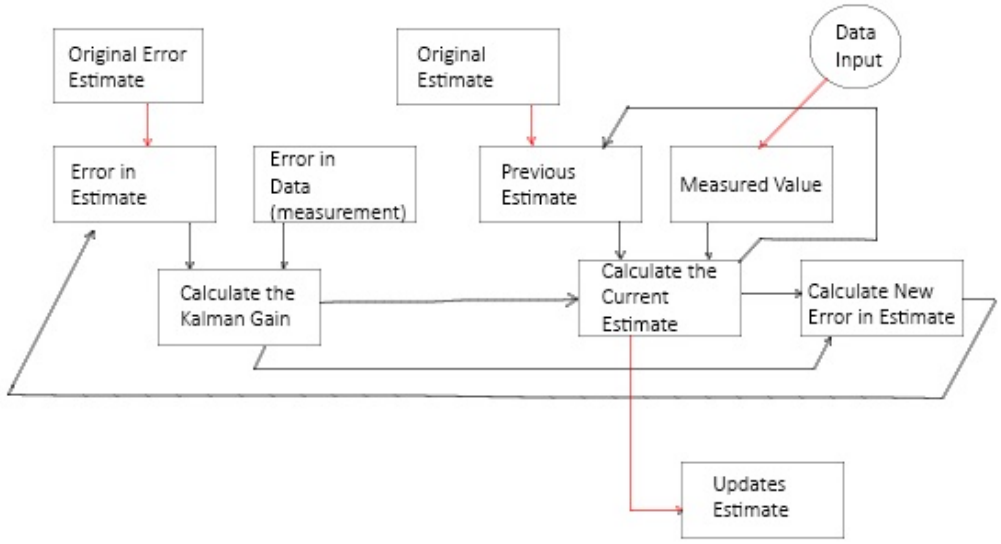


Figure 72: Complimentary Filter Working Model

It is a reliable and best-estimation algorithm that gives a prediction of the magnitude while the data are being generated. The mathematical concept behind this is to find the intersection between the linear estimation and the unbiased estimation, which is also called the best linear unbiased estimate [12].

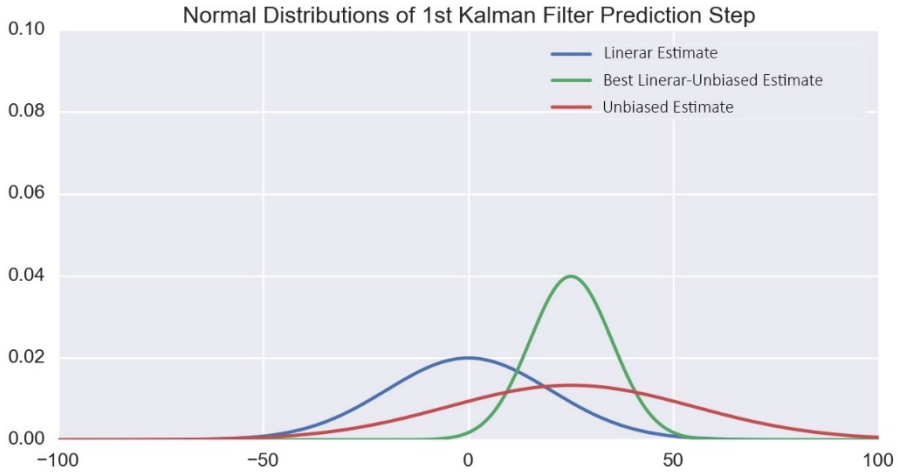


Figure 73: Kalman Estimation

The Kalman filter is a discrete time linear dynamic system based on two equations:

$$x_k = Ax_{k-1} + w_k \text{ and}$$

$$z_k = Bx_k + v_k$$

x_k refers to the state of the system at z_k. The state of the system can be described in a matrix of position, velocity and acceleration.

$$x_k = \begin{bmatrix} x \\ dx \\ d2x \end{bmatrix}$$

x_{k-1} should be the x_k in the previous state

The A is the operation used on the previous state to calculate the current state, and it is represented in a matrix through the standard physics kinematic equation.

$$A = \begin{bmatrix} 1 & 1 & 1/2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

In the second equation, B is an identity matrix. In a noiseless system, z_x should be equal to x_k. The real representation of the equation was done by adding w_k and v_k which are the Gaussian noise.

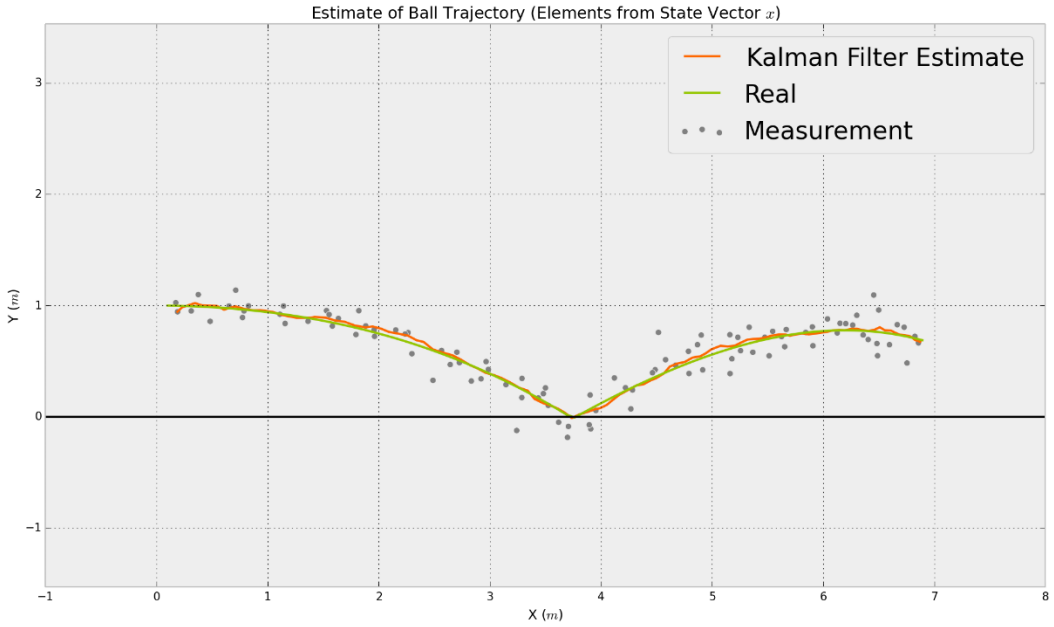


Figure 74: Kalman Estimation 2

This estimation [11] gives the true values which are closer to the actual one. The generated curve or data will be stable and error free to process further. The other estimators are the Alpha-Beta Filter, Zakai Equation and Stochastic

Differential Equation. The Kalman filter is the best among the mathematical estimators. Even the assumptions are best and comparable to the electronic approach of low pass filtering.

Sensor fusion through complementary filters

The IMU device [36] developed here contains a magnetometer, a gyroscope and an accelerometer. By giving a tweak to the Kalman filter, a complimentary filter [28] can be designed. There is a need for the complementary filter to minimise the gyro drift and the signal noise from the accelerometer and the magnetometer. It enables calculation of the error-free orientation of the device.

Getting the three orientation angles should be enough to calculate the altitude of the device. Of them, the most important are to estimate the device's acceleration and its magnetic north. The accelerometer also gives the vector that points toward the centre of the earth because it also measures the acceleration due to gravity. The magnetometer provides the precision of the moving device according to the earth's magnetic pole [11]. Both of the sensors provide sufficient information to calculate the device's orientation. But the raw information usually has inaccurate data; notably, the magnetometer data contains more noise.

The gyroscope [5] is the somewhat more accurate sensor but has the flaw of very much lower response time and suffers gyro drift. The gyro provides the rotation angular speed for the three axes. These speed values are needed to integrate at the time to get the actual orientation [28]. It means the angular speed was multiplied by the interval of time between the previous and the current sensor output. It gives the rotational increment.

The summation of the rotational increments in total yields the orientation. This process generates small errors in each step of the iteration. These errors added together result in a delay in the calculated direction. This is known as gyro drift [11].

As mentioned earlier, to avoid noisy orientation, flow and uncertainties, the gyroscope data are utilized in very limited way. These gyro data are needed for a short interval of time only when there is a change in the orientation. Whereas, the other two sensors require generating data consistently. It is equivalent to filtering. So there is a need for processing a low pass filter estimate which eliminates the higher frequencies and a high pass filter rating which removes the low frequencies at the same time. This was done by introducing the "complementary filters" which make the estimates for both and fuses the sensors together and also contributes for assuming the altitude [28] [36].

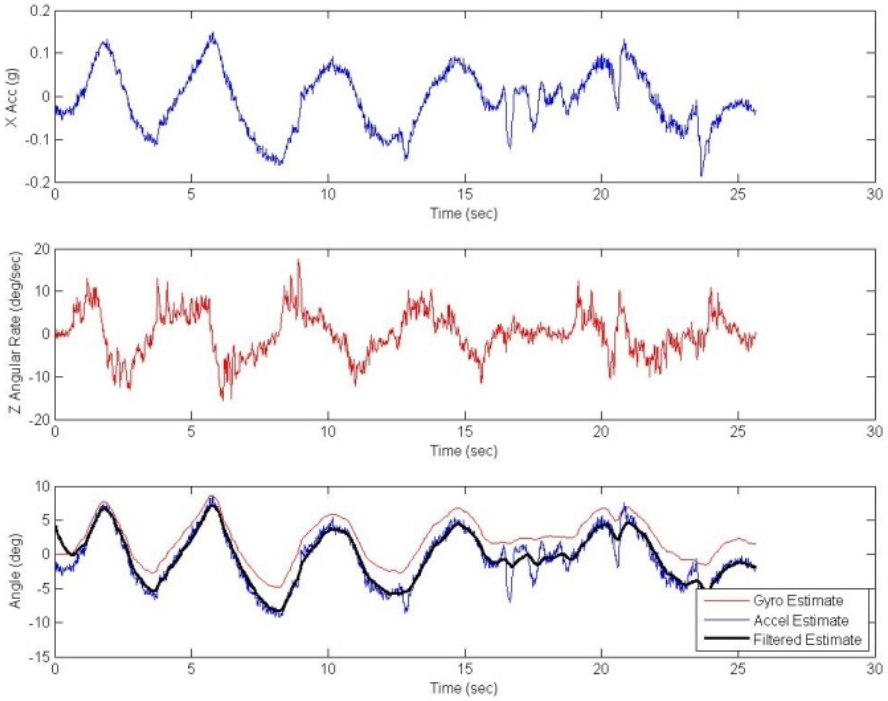


Figure 75: A Sample Graph on Sensor Fusion

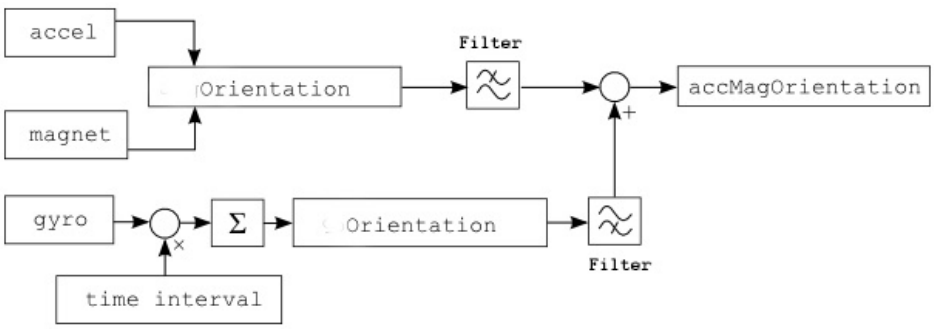


Figure 76: Orientation Filter and Sensor Fusion

The accelerometer and the magnetometer orientation should be processed through a low pass filter estimate, and the gyro orientation should be treated through the high pass filter estimate to correct the accelero-magneto orientation, which is also called the filtered estimate.

The sample graph shows how the curve which is estimated adopts values closer to the original curve, which is still uncertain by a minimum fraction [28]. Madgwick's complimentary filtering method reduces the uncertainty factors and provides the immersive experience in the virtual world without the flaws of noise and drift.

Madgwick's AHRS (Altitude Heading Referencing System) literature

The optimised measurement plays a significant role in various fields of robotics, aerospace, navigation and animation. A sensor-based inertial measuring unit is a self-contained system which does not need the external device to track the motion.

AHRS [11] is a complimentary filter algorithm which is successfully executed by Sebastian O.H. Madgwick in the year 2009/10 at the University of Bristol. The other best implementation of a complimentary filter was achieved by Robert Mahony, which is an efficient and effective solution. However, the application is only applicable for an IMU.

An IMU contains the sensors for measuring the acceleration and the gyro tilt. Adding a magnetometer in the combination will enable the IMU to have an additional edge on getting the orientation towards the earth's magnetic field. This kind of hybrid IMU is also called a MARG sensor.

An IMU can measure the altitude about the earth's gravity, but the MARG can measure the height as well as provide the orientation, which is related to the magnetic field of the Earth and the direction of the earth's gravity. The MARG system was also known as an AHRS (Altitude Heading Referencing System).

Madgwick's AHRS orientation filter is valid for both IMU and MARG sensors. The job of an orientation filter is to fuse the measurements from the gyroscope, accelerometer and magnetometer together and to compute the estimate of the orientation.

Most of the IMU-based system like Xsens, MicroStrain, VectorNav, InterSense, PNI and Crossbow use Kalman-based solutions because of their reliability and accuracy. Although the Kalman filter is familiar, it also has disadvantages in complexity, complications and requires an enormous computational load. The human or biped motion capture is the best example of it. It requires a sampling rate between 512 Hz and 30 Hz. Its challenges motivated for an alternative approach. The fuzzy processing, fixed filters and Bachman approaches are some of the brilliant attempts but these failed to produce a result. The Mahony and Madgwick methods are the best methods for the current model which is based on the Kalman approaches. The filter employs the representation called "Quaternion" which is an alternative to the "Euler" which suffers a gimbal lock.

Quaternion representation

A quaternion is a mathematics involved in locating an object in a 3D space with the four-dimensional complex numbers. It is a method of describing an object in a 3D space. The mathematician William Rowman Hamilton invented this in 1805. The object in the 3D space is also referred to as the rigid body or the coordinate frame. It is done by calculating the current position with the previous position. For an instant, frame A is the initial position and Frame B is a new position. An arbitrary orientation of frame B is achieved about the angle it makes to the frame A. The mutual orthogonal units to represent frames A and B are vectors (X_A, Y_A, Z_A) and (X_B, Y_B, Z_B.). It defines the principal angles respectively.

Equation i^2- j^2- k^2- ijk -1 is the general representation of the quaternion. Madgwick used the quaternion mathematics for all his iterations. His model received inputs as raw data from the magnetometer, accelerometer and

gyroscope and return the values as a quaternion notation (q0, q1, q2 and q3) after applying the filter.

Madgwick Method

- Derive the orientation from the angular rate
- Derive the orientation from the vector observations
- Implement the fusion filter algorithm
- Use the magnetic distortion compensation
- Use the gyroscope drift compensation
- Apply the filter gains.

Madgwick posted his work in Google Code in 2010. His code has been taken from http://www.x-io.co.uk/open-source-imu-and-ahrs-algorithms/ and adapted for an Arduino form that suits the current device. The class MadgwickAHRS.h holds his methods, which are described above. Coding is in Appendix 3 in Madgwick's Algorithm.

Quaternions are the representation used here for rotating a 3D object in Unity. This has been because Euler angles have a gimbal lock issue. So a method was needed for adaptation inside the Madgwick.h to receive the data in the form of quaternions.

The code is altered such that it return quaternions only, using the following method:

```
filter.updateIMU(gx / factor, gy / factor, gz / factor, ax, ay, az);
```

From the above line, the *update* function fetches the filtered data from the sensors.

The function uses these variables to get the quaternion to the Main program:

```
float getPitch(){return atan2(2 * q2 * q3 - 2 * q0 * q1, 2 * q0 * q0 + 2 * q3 * q3 - 1);};
float getRoll(){return -1 * asin(2 * q1 * q3 + 2 * q0 * q2);};
float getYaw(){return atan2(2 * q1 * q2 - 2 * q0 * q3, 2 * q0 * q0 + 2 * q1 * q1 - 1);};
```

However, the above lines have a *get* function that converts the quaternion into Yaw, Pitch and roll which are not of interest here.

So, the following code is used to get the quaternion from the library:

```
float getVecA(){return q0; };
float getVecB(){return q1; };
float getVecC(){return q2; };
float getVecD(){return q3; };
```

These lines of code get the constant value which was recently generated by the algorithm.

On the Arduino side, the quaternions are called by declaring variables:

```
float vecA;
float vecB;
float vecC;
float vecD;
```

Through the variables the following are called:

```
vecA = filter.getVecA();
vecB = filter.getVecB();
vecC = filter.getVecC();
vecD = filter.getVecD();
```

The quaternion was printed to the serial monitor with a delay of 15 milliseconds per data. The received data were tested graphically using Unity [22].

Madgwick implemented his method on both IMU and the MARG and tested them. Now the task was to test the algorithm in the HMC5883L and MPU6050 based MARG device [11].

Arduino program

The file MadgwickAHRS.h was copied into the Arduino library folder and the following program loaded to the Arduino.

Step1: Load the libraries

```
#include "I2Cdev.h"
#include "MPU6050.h"
#include "HMC5883L.h"
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
#include "Wire.h"
#endif
#include "MadgwickAHRS.h"
```

Step2: Load the devices

```
MPU6050 accelgyro;
HMC5883L mag;
```

Step3: Declare the variables

```
Madgwick filter; // initialise Madgwick object
int16_t ax, ay, az;
int16_t gx, gy, gz;
int16_t mx, my, mz;
float yaw;
float pitch;
float roll;
float vecA;
float vecB;
float vecC;
float vecD;
int factor = 800; // variable by which to divide gyroscope values, used to control sensitivity
// note that an increased baud rate requires an increase in value of factor
int calibrateOffsets = 1; // int to determine whether calibration takes place or not
```

Step4: Initialize I^2C implementation, serial and the devices

```
void setup() {
 #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
      Wire.begin();
  #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
      Fastwire::setup(400, true);
  #endif
 Serial.begin(9600);
 accelgyro.initialize();
 mag.initialize();
}
```

Step5: Read the raw values and process them

```
void loop() {
// read raw accel/gyro/Magnetometer measurements from device
accelgyro.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
mag.getHeading(&mx, &my, &mz);
// use function from MagdwickAHRS.h to return quaternions
 filter.updateIMU(gx / factor, gy / factor, gz / factor, ax, ay, az);
vecA = filter.getVecA();
vecB = filter.getVecB();
vecC = filter.getVecC();
vecD = filter.getVecD();
// alternate functions to find yaw roll and pitch from quaternions
 /* yaw = filter.getYaw();
     roll = filter.getRoll();
     pitch = filter.getPitch();*/
```

Step5: Print the processed values

```
//Print Quaternions
    Serial.print(vecA);
    Serial.print(","); // print comma so values can be parsed
    Serial.print(vecB);
    Serial.print(","); // print comma so values can be parsed
    Serial.print(vecC);
    Serial.print(","); // print comma so values can be parsed
    Serial.println(vecD);
    delay(15);
//Print Yaw, Pitch and Roll
/* Serial.print(yaw);
    Serial.print(","); // print comma so values can be parsed
    Serial.print(pitch);
    Serial.print(","); // print comma so values can be parsed
    Serial.println(roll);
    delay(15);*/
}
```

Unity program

In Unity, the C# code over the game object:

- transform.rotation = Quaternion.Slerp(transform.rotation, new Quaternion(float.Parse(vec3[0]),float.Parse(vec3[1]),float.Parse(vec3[2]), float.Parse(vec3[3])),0.20f);

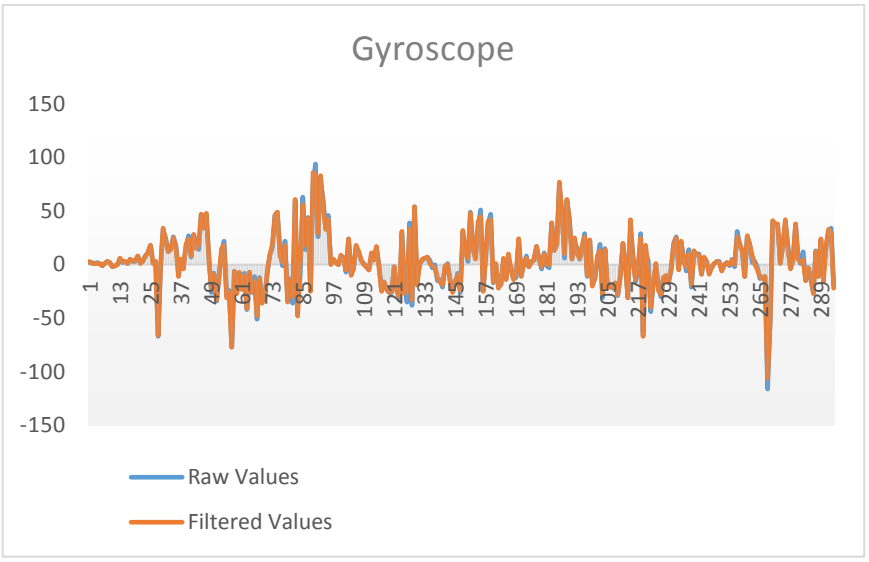
takes input from the four values of the quaternion and rotates the object:

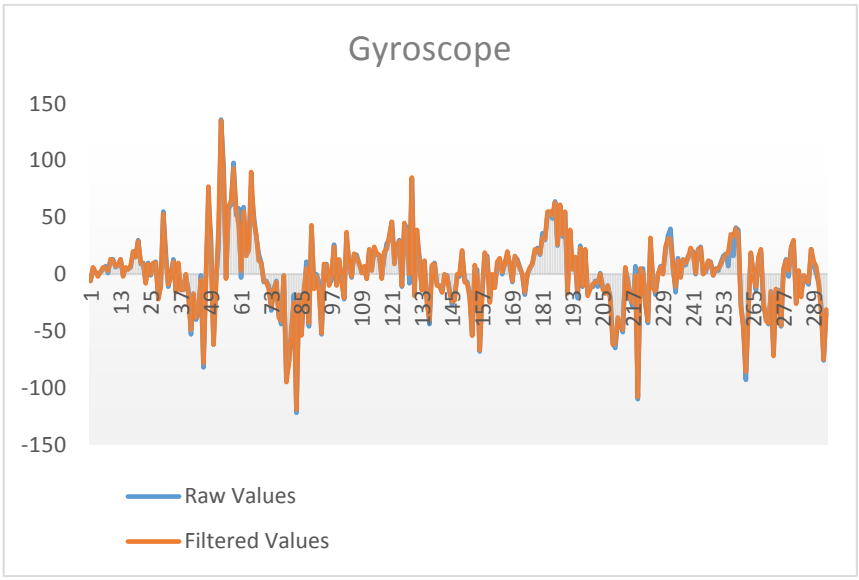
- rot = new Vector3(float.Parse(vec3[0]),float.Parse(vec3[1]),float.Parse(vec3[2])); transform.rotation=Quaternion.Slerp(transform.rotation,Quaternion.Euler(rot.x,rot.y,rot.z),Time.deltaTime*3)

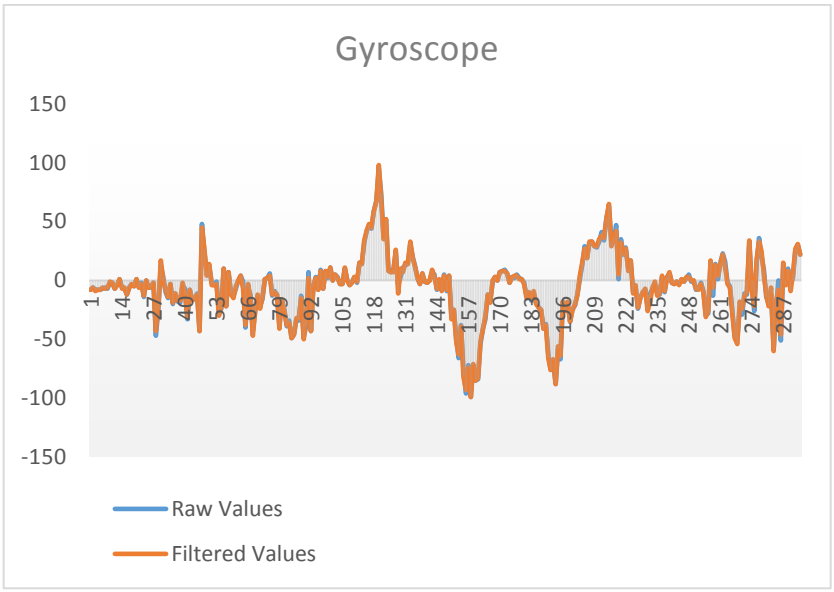
And this takes input from the yaw, pitch and roll and rotates the object accordingly. The translation is also done using the same method.

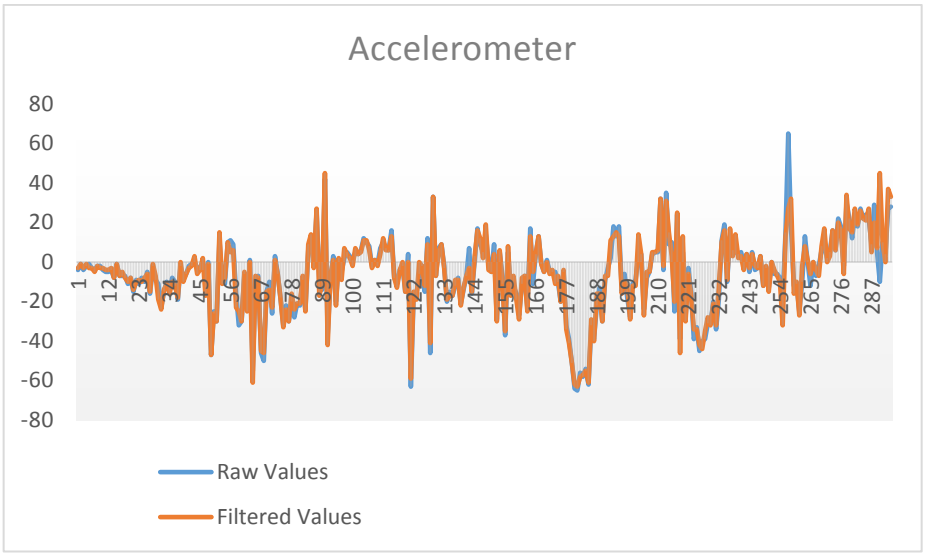
Experimentation

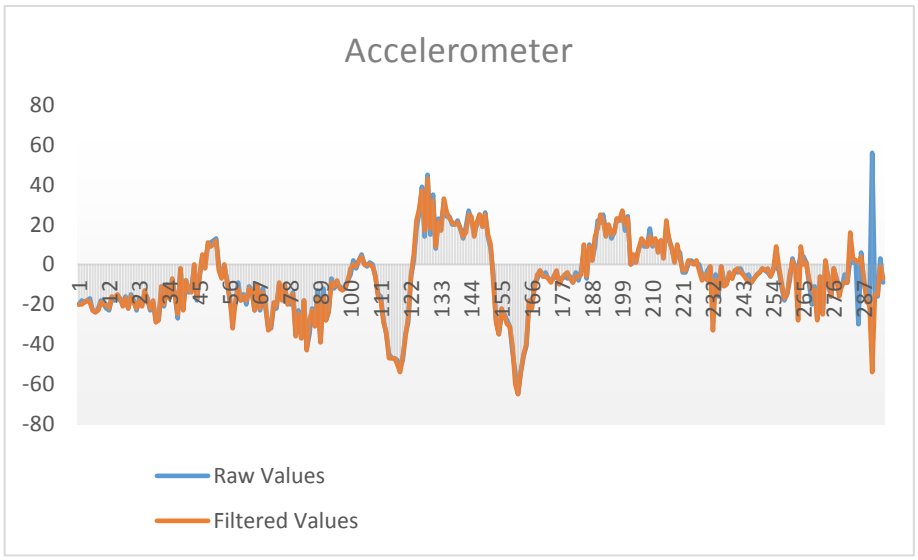
An experiment is done studying the estimated and the original sensor values. For the raw data, the values were printed without processing the filter. The Arduino's Ethernet shield was used and the memory card was used to record the data. A random "L" movement is made in all axes while recording the data. The breaking point of the magnitude is the focus of the experimentation.


Gyroscope
150
100
50
0
-50
-100
-150
1
13
25
37
49
61
73
85
97
109
121
133
145
157
169
181
193
205
217
229
241
253
265
277
289
Raw Values
Filtered Values


Gyroscope
150
100
50
0
-50
-100
-150
1
13
25
37
49
61
73
85
97
109
121
133
145
157
169
181
193
205
217
229
241
253
265
277
289
Raw Values
Filtered Values


Gyroscope
150
100
50
0
-50
-100
-150
1
14
27
40
53
66
79
92
105
118
131
144
157
170
183
196
209
222
235
248
261
274
287
Raw Values
Filtered Values


Accelerometer
80
60
40
20
0
-20
-40
-60
-80
1
12
23
34
45
56
67
78
89
100
111
122
133
144
155
166
177
188
199
210
221
232
243
254
265
276
287
Raw Values
Filtered Values


Accelerometer
80
60
40
20
0
-20
-40
-60
-80
1
12
23
34
45
56
67
78
89
100
111
122
133
144
155
166
177
188
199
210
221
232
243
254
265
276
287
Raw Values
Filtered Values

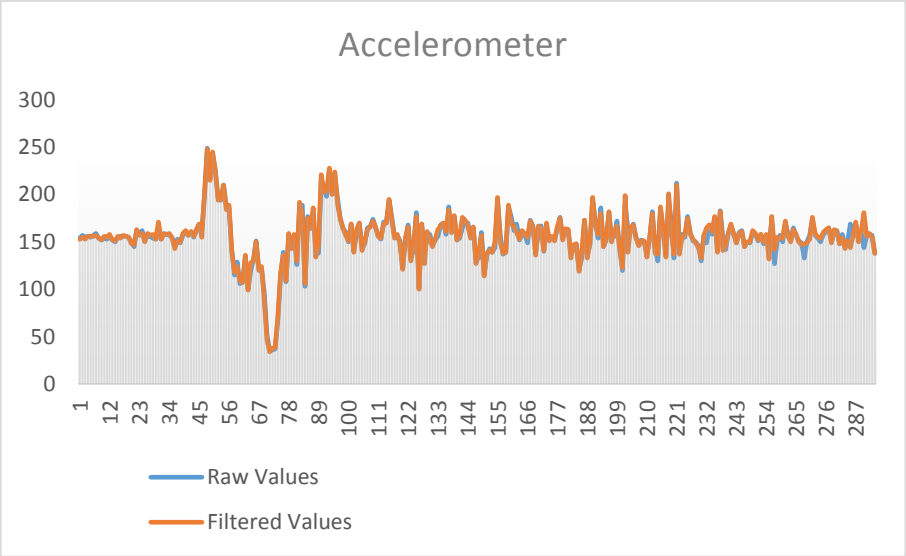


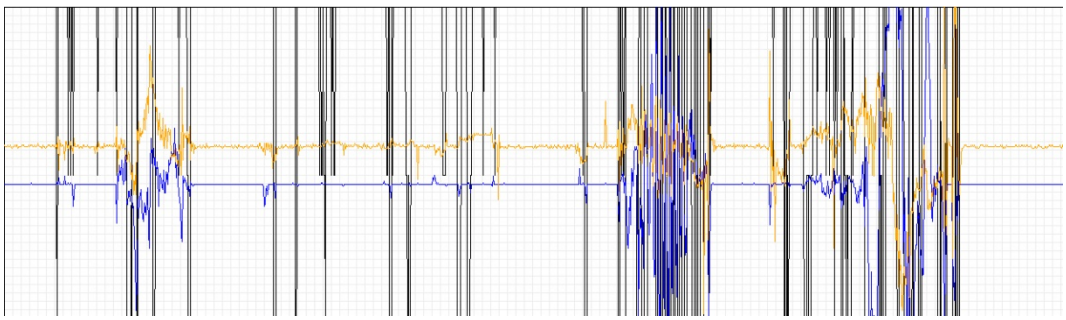
Figure 77: Experimentation Gyroscope and Accelerometer

The graphs above were drawn in PowerPoint for the data received on the serial port.

Live test

Analysis software was also used to test it live. These are the live results from the device that give the graphs with and without the filter [11]. The program was altered to print both the raw and filtered values to receive both values simultaneously. A demonstration was made to present the sensor data while at rest and when it is disturbed.

- The graph below shows the gyro-compensation.



- The chart below shows the accelero-compensation.

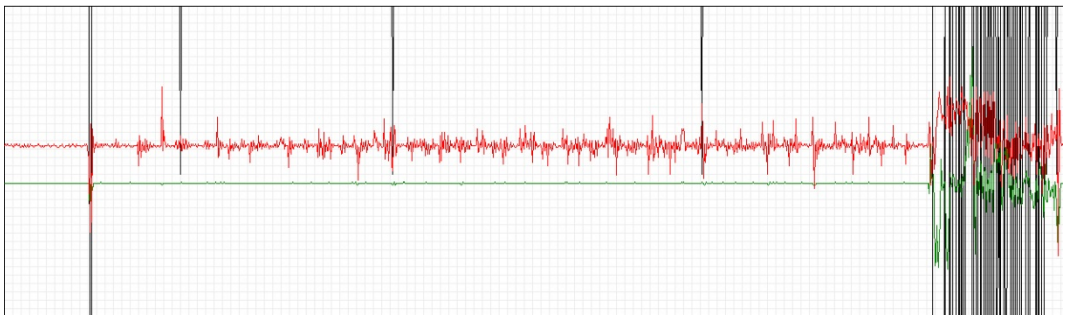


Figure 78: Experimentation real-time Graph

Evaluation

An evaluation was conducted in one of the university's labs on March 10th, 2016. The assessment was about how people feel about using the motion controller apart from the other traditional game controllers.

At this stage, the cricket simulation could not be used because this study was primarily about involving a joystick and a keyboard as the game controller. The work on other game artefacts was to further development understanding of the software pipeline between motion capture and game production, as noted in the Context.

This evaluation was a qualitative comparison between the motion controller, the joystick and the keyboard controller.

Data were collected using the SurveyMonkey mobile application.[3]

Simulation for the evaluation

A simple game was made to visit a studio flat in a spying drone. The drone was controlled by the all of these controllers. The students should play the game with these drivers and evaluate them. The evaluation was an online survey that consisted of nine multiple-choice questions, as shown in the following figures.

[3] https://www.surveymonkey.com/mp/surveymonkey-app/

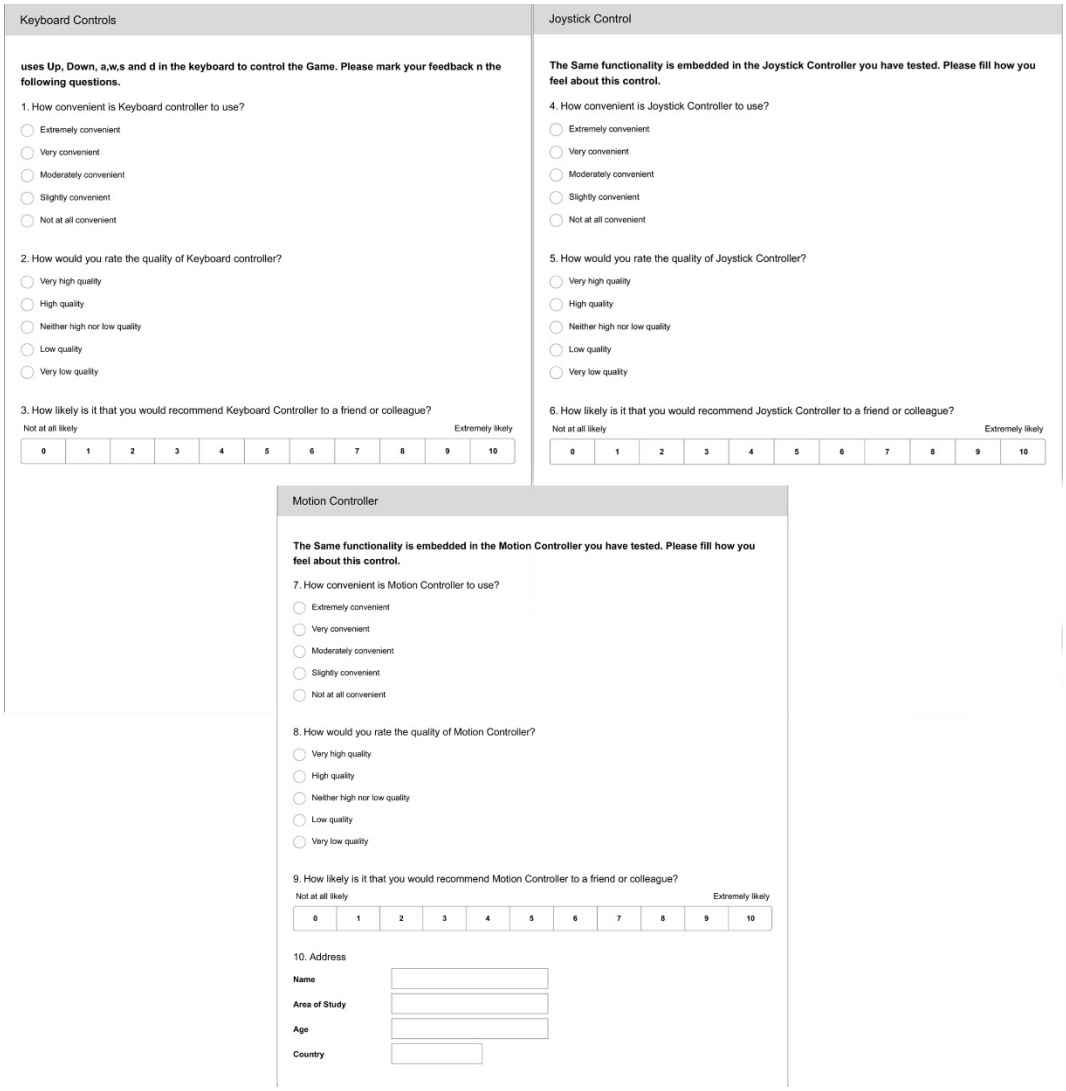


Figure 79: Evaluation 3 Test

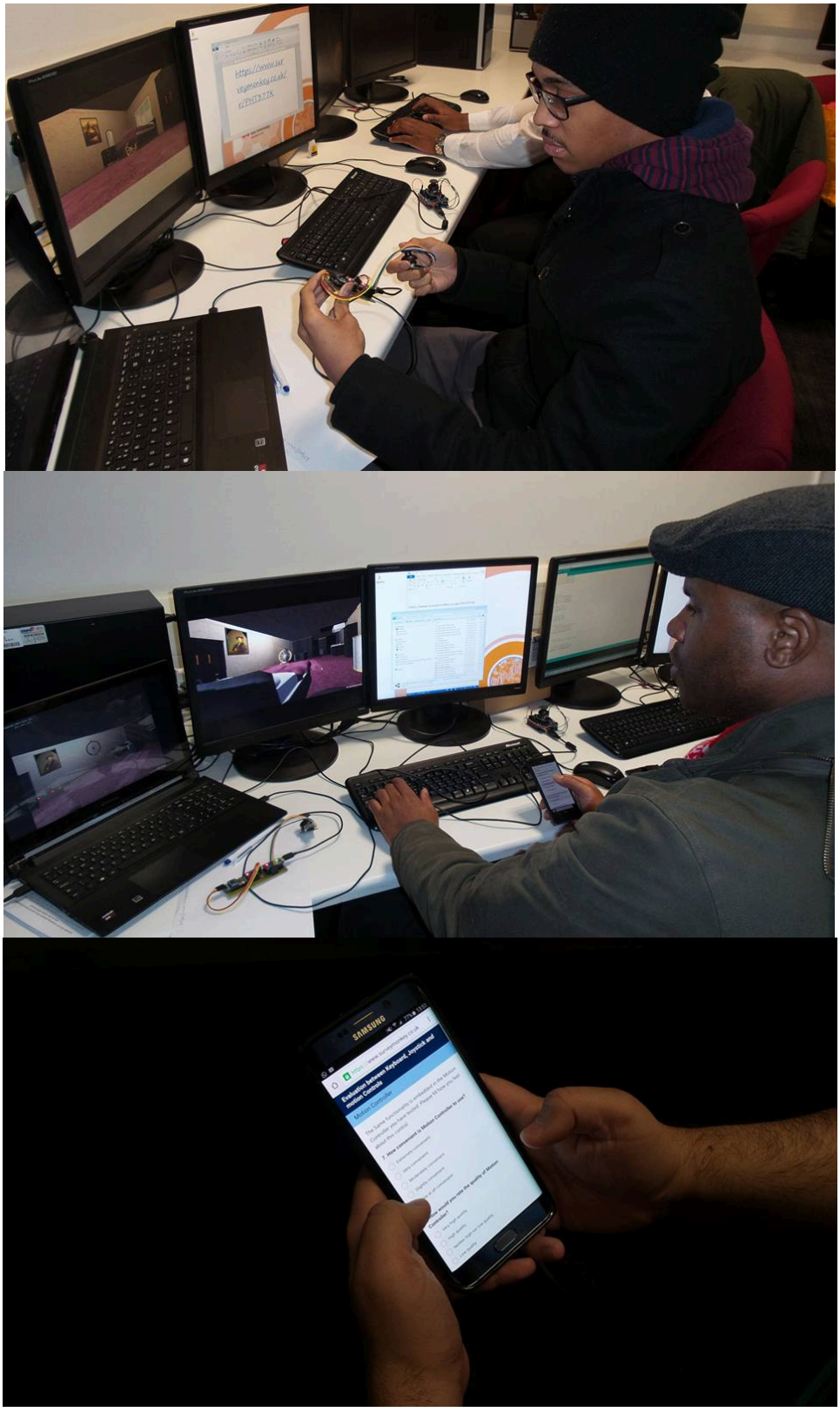
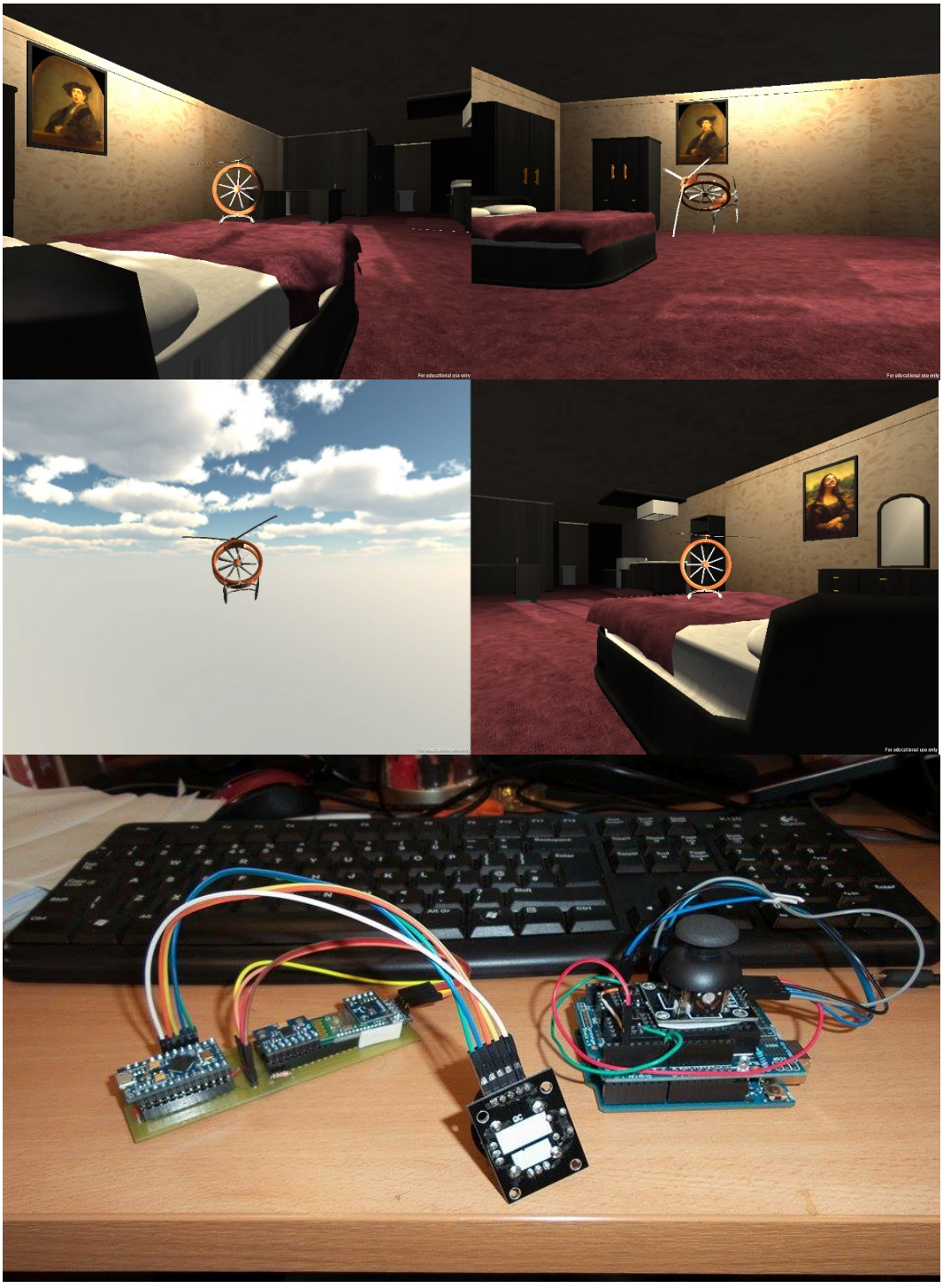


Figure 80: Evaluation

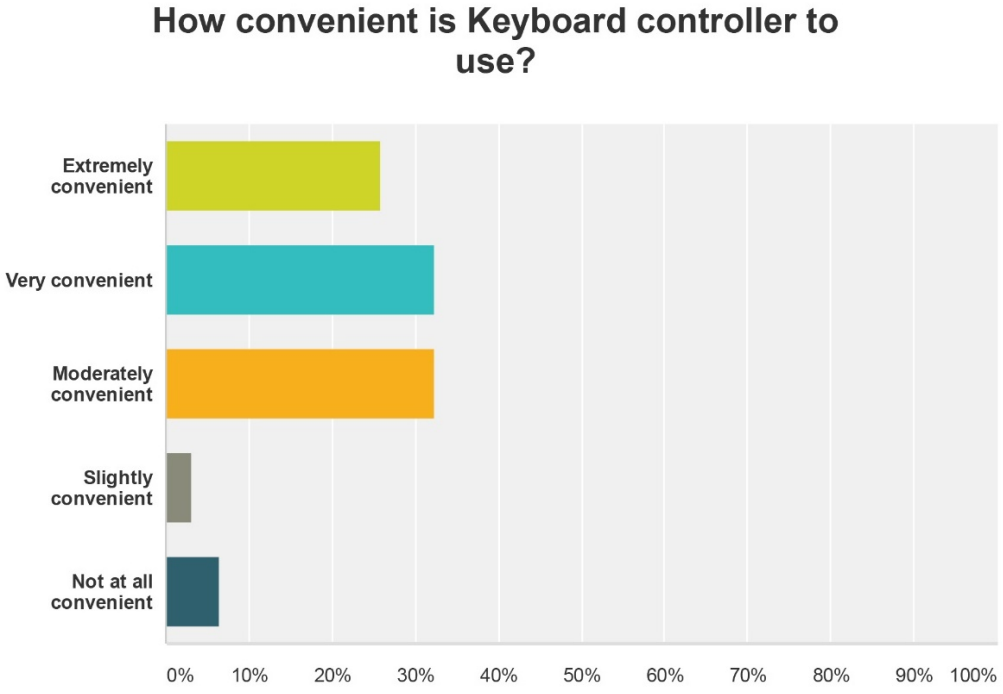


All the gaming assets were created for this simulation:

- 3D models created using Maya
- Animation done in Maya
- Game developed in Unity 3D
- Motion controllers and joystick are accomplished using the Arduino

Evaluation of convenience

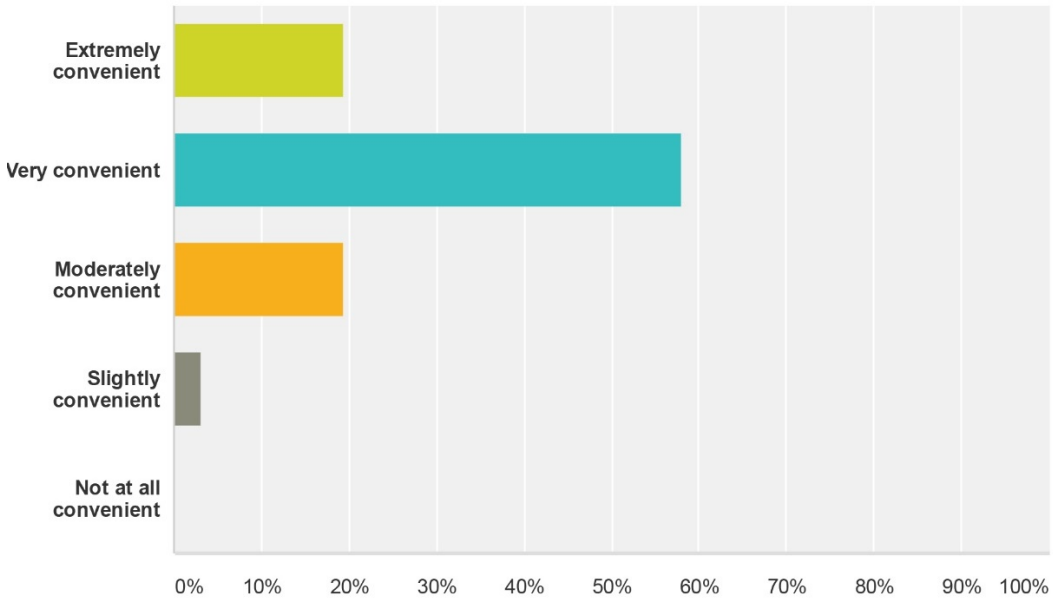
The analysis was made to check the convenience factor between the controllers**.**



Answer Choices	Responses	
Extremely convenient	**25.81%**	8
Very convenient	**32.26%**	10
Moderately convenient	**32.26%**	10
Slightly convenient	**3.23%**	1
Not at all convenient	**6.45%**	2
Total		31

How convenient is Joystick Controller to use?

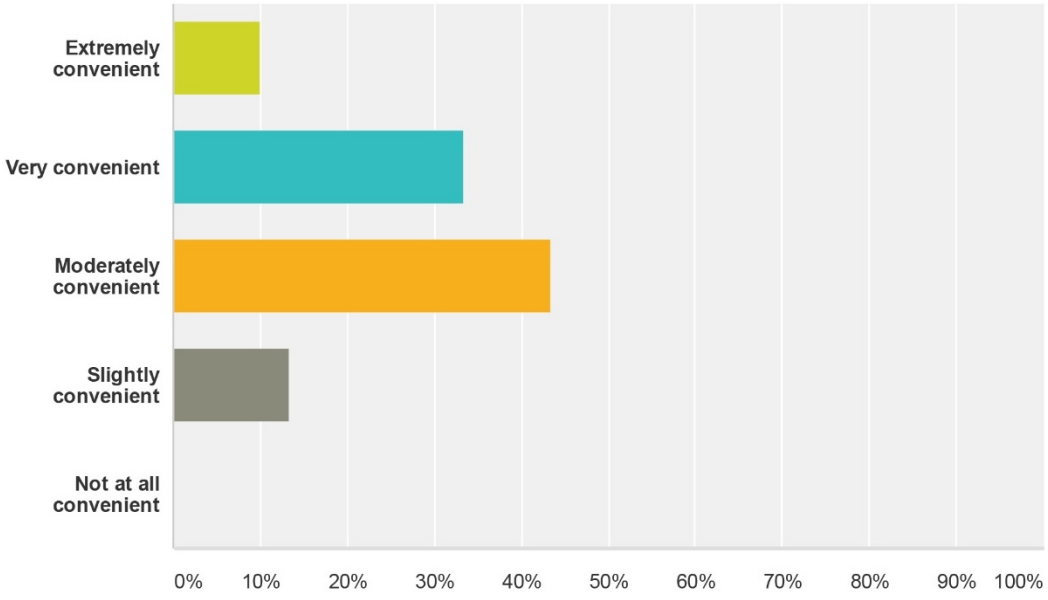
Answered: 31 Skipped: 1



Answer Choices	Responses	
Extremely convenient	**19.35%**	6
Very convenient	**58.06%**	18
Moderately convenient	**19.35%**	6
Slightly convenient	**3.23%**	1
Not at all convenient	**0.00%**	0
Total		31

How convenient is Motion Controller to use?

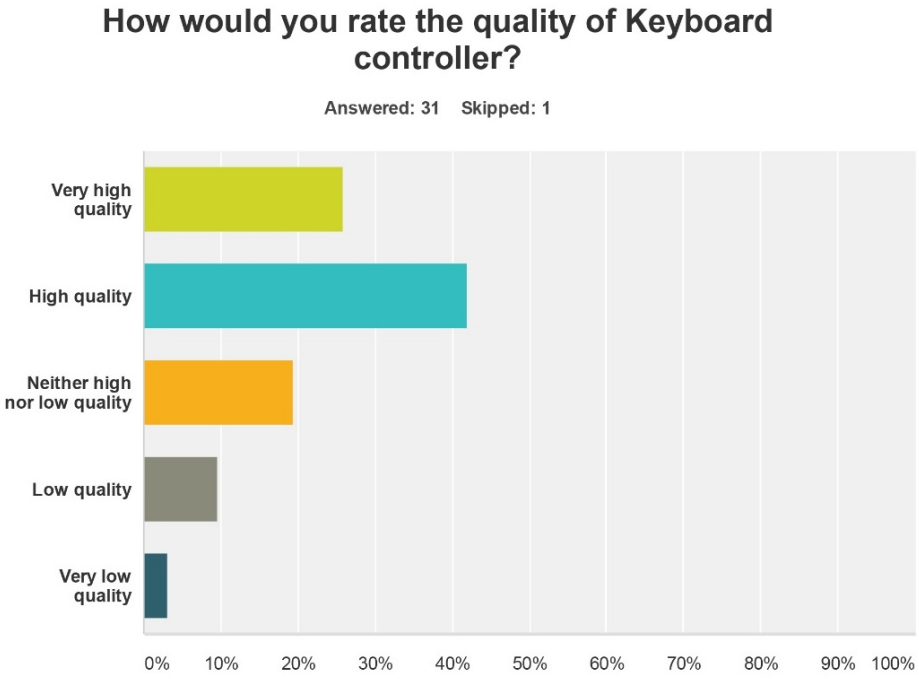
Answered: 30 Skipped: 2



Extremely convenient	10.00%	3
Very convenient	33.33%	10
Moderately convenient	43.33%	13
Slightly convenient	13.33%	4
Not at all convenient	0.00%	0
Total		30

Evaluation of quality

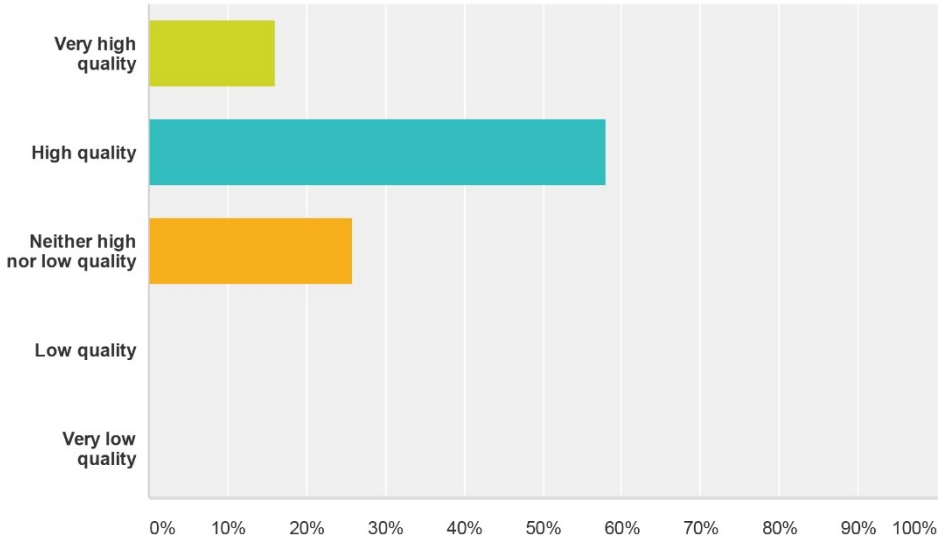
The analysis was made to check the quality between the controllers**.**



Answer Choices	Responses	
Very high quality	**25.81%**	8
High quality	**41.94%**	13
Neither high nor low quality	**19.35%**	6
Low quality	**9.68%**	3
Very low quality	**3.23%**	1
Total		31

How would you rate the quality of Joystick Controller?

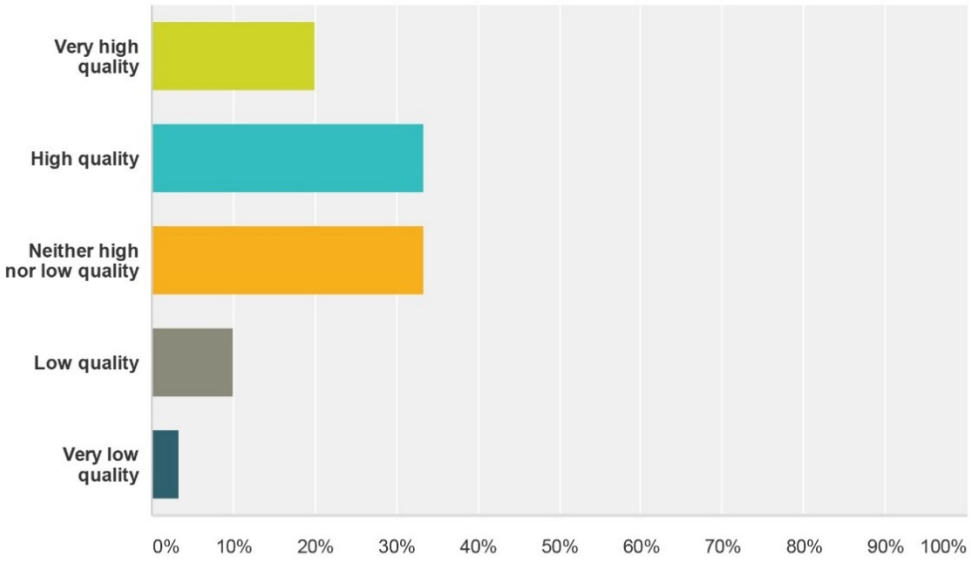
Answered: 31 Skipped: 1



Answer Choices	Responses	
Very high quality	**16.13%**	5
High quality	**58.06%**	18
Neither high nor low quality	**25.81%**	8
Low quality	**0.00%**	0
Very low quality	**0.00%**	0
Total		31

How would you rate the quality of Motion Controller?

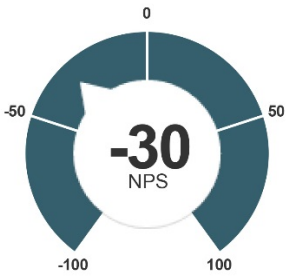
Answered: 30 Skipped: 2



Answer Choices	Responses	
Very high quality	20.00%	6
High quality	33.33%	10
Neither high nor low quality	33.33%	10
Low quality	10.00%	3
Very low quality	3.33%	1

How likely is it that you would recommend Keyboard Controller to a friend or colleague?

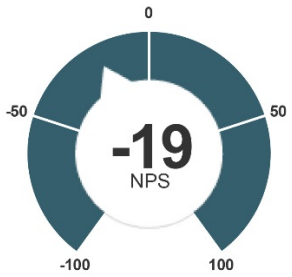
Answered: 30 Skipped: 2



Detractors (0-6)	Passives (7-8)	Promoters (9-10)	Net Promoter® Score
47% 14	37% 11	17% 5	-30

How likely is it that you would recommend Joystick Controller to a friend or colleague?

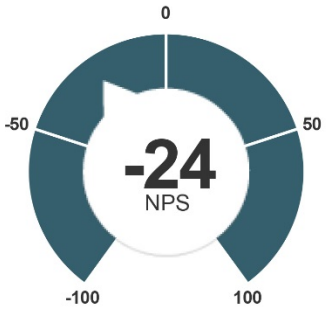
Answered: 27 Skipped: 5



Detractors (0-6)	Passives (7-8)	Promoters (9-10)	Net Promoter® Score
33% 9	**52%** 14	**15%** 4	**-19**

How likely is it that you would recommend Motion Controller to a friend or colleague?

Answered: 29 Skipped: 3



Detractors (0-6)	Passives (7-8)	Promoters (9-10)	Net Promoter® Score
48% 14	**28%** 8	**24%** 7	**-24**

Conclusion

In conclusion, the project's aim was to design an approach to VR game control suitable for a mobile platform. The problem this is intended to solve is that current gameplay in a mobile VR cannot satisfy the complete VR experience and is also unsafe to play while the mobile is constantly in contact with the head.

The proposed solution adapts Google Cardboard technology which uses the sensors in the mobile to track the head and uses the phone screen as a display. The gameplay uses a normal joystick to control the game.

The proposed has a HMD to connect with the phone through the OTG as a serial device. It is plugged to play like a hands-free. The HMD is free from any camera that tracks the head. The HMD has sensors to track the movement.

The design uses the phone as the game controller which is superior to the regular joystick because the user can use the phone as a motion controller which should be the most compatible controller for a mobile VR game and also give a live capture of the body movement.

A specialized motion controller is provided with joystick in this project, which is useful to avoid the health risks of handling the phone too much, and also with the intention of using it as a left-hand controller to track the other hand.

Outcomes and assessment

The objectives of the study were:

1. To assess existing VR game controls
2. To design a VR game control suitable for a mobile platform
3. To test the mobile VR game control
4. To evaluate the proposed design against existing designs

The idea initially desired was a VR game control using speech and motion tracking [3]. But within this project only the motion controls could be accomplished with an HMD.

Animation technology, motion capture technology and the pipeline to produce a game using Unity and other production software has been explored deeply here. The university's department of interactive technologies environment provided the knowledge for creating the game from scratch, which provides context for developing the gaming experience.

The researcher has taken the opportunity presented by teaching students to compared with them animation and motion capture and methods of motion capture. Their views were collected in a survey session.

Comparison of animation with motion capture

The response is mixed, the students believed that this would make the work easy and also believed this could cost more for production in the same time. They required a simpler solution at an affordable price.

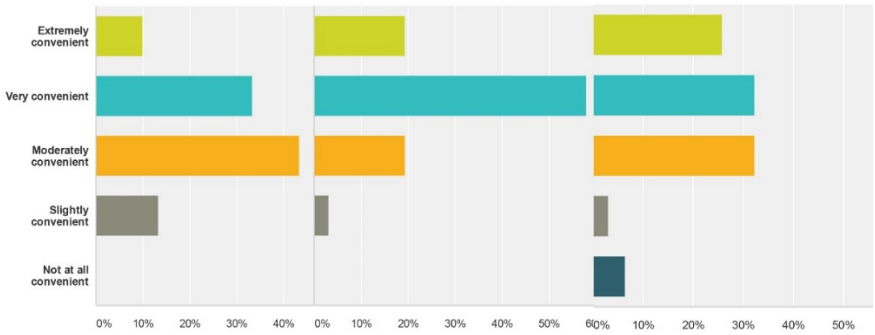
Comparison of optical and non-optical motion capture

The response is favourable towards the non-optical motion capture (proposed motion controller). This could bring the price down. So that it can also be used to produce animation commercially.

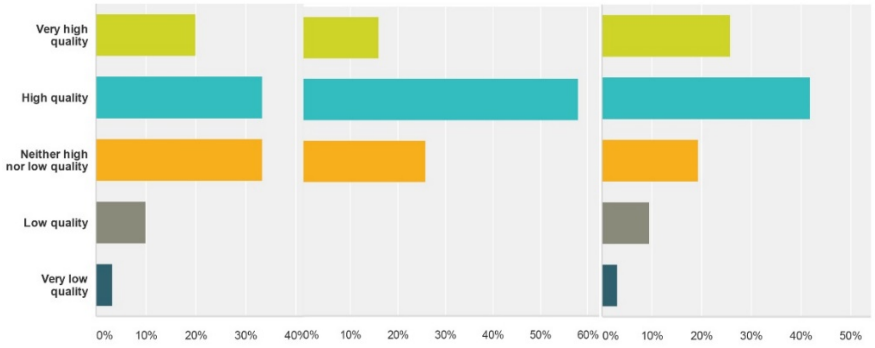
This study proves that there are requirements and demands for the research presented.

The system was evaluated by asking some students to act as users to test it. The evaluation compared the proposed motion controllers with the other controllers.

Convenience and Quality chart:



Motion Controller Joystick Controller Keyboard Controller



The results are moderately positive and prove the proposed motion controllers are more convenient than the traditional keyboard controls.

This concludes that the system proposed here was a good attempt at achieving the objectives.

The researcher has presented this work at the University's Postgraduate Annual Winter Conference 2015.

The researcher is also proud that the work was nominated for the Biztech BrightSparc Award 2016 where he was runner-up in the student category for the best innovation award.

Future works

In a VR game, the conversations, dramas and characters which the players bring to the field should give an immersive experience to the game more than playing with controls. Achieving voice-triggered animation must have a greater impact on the experience [8], as will real-time facial animation between avatars participating in the virtual world. Adding all these VR aspects to a sporting game will take the gaming experience to the next level.

Future works in VR are required for achieving all five senses in the virtual world in order to improve the immersiveness. The areas are:

1. Interactive stereo visual
2. Binaural hearing

3. Interactive speech
4. Electro-physical touch-feedback
5. Smell of the scene

The mechanical method [26] [27], for instance, an exoskeleton tracking that involves many potentiometers, can also give an entire track. The Gypsy motion capture suite and Dexmo F2 are existing systems which work on this principle. Particularly, Dexmo F2 can provide the electro-physical touch-force feedback, which is impressive.

Modern commercial products [24] such as Salto and GloveOne use the flexi-sensors along with other sensors that generate data according to the threshold of the phone material which is bent. The Myo armband collects data from muscle contraction and senses the movement about to be made and produces the track accordingly. This works under a logic of recognising the gesture with the signals from the forearm muscles. The Myo also gives a clue for future innovation to determine the physical motion directly from the brain.

If VR adopts this technology, humans can live virtually and their brain is connected to the virtual world like in a science fiction movie.

References

[1] G. Cacho, “Are VR headsets like Project Morpheus and Oculus Rift the future of gaming?,”*A+E Interactive* [Online], March 27, 2014. http://blogs.mercurynews.com/aei/2014/03/27/are-vr-headsets-like-project-morpheus-and-oculus-the-future-of-gaming/

[2] T. Parisi, *Learning Virtual Reality: Developing Immersive Experiences and Applications for Desktop, Web, and Mobile*, USA: O’Reilly Media , 2015.

[3] A. Covert, “Video game controllers need innovation,” *CNN Money* [Online], June 14, 2013. http://money.cnn.com/2013/06/14/technology/innovation/motion-control-games/

[4] C. Roquilly, “Control over virtual worlds by game companies,” *Management Information Systems Quarterly* 35(3), 653-671, , 2011.

[5] K. B. Lee, *Principles of MEMS*, Hoboken, NJ: Wiley, 2010.

[6] A. Menache, *Understanding Motion Capture for Computer Animation*, 2nd ed. Elsevier, 2011.

[7] *Web3D '14 The 19th International Conference on Web3D Technology, Vancouver, BC, Canada — August 08 - 10, 2014*, New York, NY: ACM, 2014.

[8] D. Freitas & G. Kouroupetroglou, “Speech technologies for blind and low vision persons”, *Technology and Disability*, 20(2), 135-156, 2008.

[9] L. Calandruccio &. H. Zhou, “Increase in speech recognition due to linguistic mismatch between target and masker speech: monolingual and simultaneous bilingual performance,” *Journal of Speech, Language and Hearing Research*, 57(3):1089-97, 2014. doi: 10.1044/2013_JSLHR-H-12-0378

[10] S. H. Kurniawan & A. J. Sporka, “Vocal interaction,” Proceedings CHI EA '08 CHI '08 Extended Abstracts on Human Factors in Computing Systems, New York: ACM, 2407-2410, 2008.

[11] S. O. H. Madgwick, “An efficient orientation filter for inertial and inertial/magnetic sensor arrays” Report. Google Code, 2010.

[12] R. E. Kalman, A new approach to linear filtering and prediction problems, *Journal of Basic Engineering*, 82 (Series D), 35-45, 1960.

[13] A. Pipes, *Drawing for Designers: Drawing skills, Concept sketches, Computer systems, Illustration, Tools and Materials, Presentations, Production techniques*, London: Laurence King, 2007.

[14] T. Palamar with L. Lanier & A. Honn, *Mastering Autodesk Maya 2013*, Indianapolis, IN: John Wiley & Sons, 2012.

[15] P. J. Naas, *Autodesk Maya 2013 Essentials*, Indianapolis, IN.: John Wiley & Sons, 2012.

[16] A. Watkins, *Getting Started in 3D with Maya*. Waltham, MA: Focal Press, 2012.

[17] J. Amin, *Beginner's Guide to Character Creation in Maya*, Worcester, UK: 3DTotal Publishing, 2015.

[18] P. Wells & J. Quinn, *Basics Animation 03: Drawing for Animation*, New York NY: AVA Publishing, 2008.

[19] S. Greenberg, S. Carpendale, N. Marquardt & B. Buxton, *Sketching User Experiences: The Workbook*, Burlington, MA: Morgan Kaufmann, 2011.

[20] D. Lavender, *Maya Manual*, London: Springer, 2003.

[21] D. Derakhshani, *Introducing Autodesk Maya 2013*, Indianapolis, IN: John Wiley & Sons, 2012.

[22] M. Smith & C. Queiroz, *Unity 4.x Cookbook*, Birmingham, UK: Packt, 2013.

[23] M. Kitagawa & B. Windsor, *MoCap for Artists: Workflow and Techniques for Motion Capture*, Abingdon, UK: Focal Press, 2008.

[24] M. Kitagawa. & B. Windsor, *MoCap for Artists*, Amsterdam & Abingdon: Elsevier & Focal Press, 2008.

[25] Motion Analysis Corporation, *Cortex 1.0 User's Manual*, Santa Rosa, CA: 2008.

[26] D. O'Sullivan & T. Igoe, *Physical Computing*, Boston, MA: Thomson, 2004.

[27] T. Igoe, *Making Things Talk: Using Sensors, Networks, and Arduino to see, hear, and feel your world*, 2nd edn., San Francisco, CA: Maker Media, 2011.

[28] M Euston, P. Coote, R. Mahoney, J. Kim & T. Hamel, "A complementary filter for attitude estimation of a fixed-wing UAV with a low-cost IMU". In 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 340-345 . doi: 10.1109/IROS.2008.4650766

[29] Motion Analysis Corporation, *Setting up Calcium and solving Marker Data*, Santa Rosa, CA, 2007.

[30] Autodesk Motion Builder 2013, Tutorials, Document version: 2012.03.32.01, 2012.

[31] M. Margolis, *Arduino Cookbook*, Sebastopol, CA: O'Reilly, 2012.

[32] M. Banzi & M. Shiloh, *Getting Started with Arduino: The Open Source Electronics Prototyping Platform*, 3rd edn., San Francisco, CA: Maker Media, 2014.

[33] "Arduino - Home", *Arduino.cc*, 2016. [Online] Available: http://www.arduino.cc. [Accessed: 15 Mar 2016].

[34] W. Goldstone, *Unity 3.x game development essentials: game development with C# and Javascript*, Birmingham, UK: Packt, 2011.

[35] H. Gsoedl, "New Multimedia Interface MHL™ Market Status and Technology," [Slides], Rohde & Schwarz GmbH & Co. KG, 2012.

[36] P. Lawitzki, "Android Sensor Fusion Tutorial - CodeProject", *Codeproject.com*, 2014. [Online]. Available: http://www.codeproject.com/Articles/729759/Android-Sensor-Fusion-Tutorial. [Accessed: 15 Mar 2016].

[37] "How the Wii Remote Works", *YouTube*, 2016. [Online]. Available: https://www.youtube.com/watch?v=ETAKfSkec6A. [Accessed: 15 Mar 2016].

[38] "iPhone 4 Gyroscope", *YouTube*, 2016. [Online]. Available: https://www.youtube.com/watch?v=YrrsSKI64vk. [Accessed: 15 Mar 2016].

[39] Y. Cai, Y. Zhao, X. Ding & J. Fennelly, Memsic Electronic Products www2.electronicproducts.com, February 2012.

[40] D. S. Ward K. R. Evenson, A. Vaughn, A. B. Rodgers & R. P. Troiano, "Accelerometer use in physical activity: best practices and research recommendations." *Medicine and Science in Sports and Exercise* 37(11 Suppl), S582-8, 2005.

[41] "ProfWaterLewin's channel", *YouTube*, 2016. [Online]. Available: https://www.youtube.com/user/ProfWaterLewin. [Accessed: 15 Mar 2016].

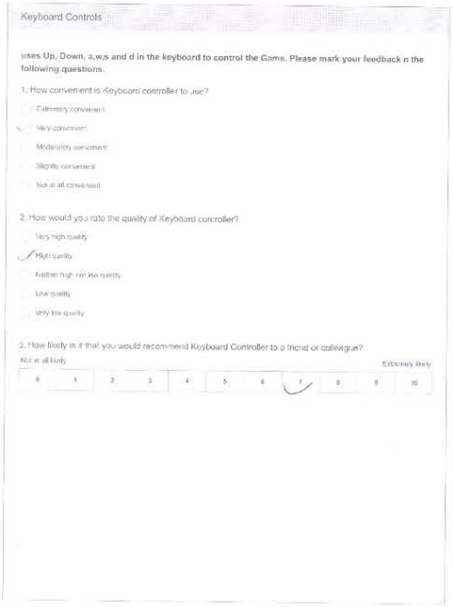
[43] "control de led con arduino leonardo + modulo bluetooth", *YouTube*, 2016. [Online]. Available: https://www.youtube.com/watch?v=45CQ2m-9FiY. [Accessed: 15Mar 2016].

[44] "Unity - Game Engine", Unity3d.com, 2016. [Online]. Available: https://unity3d.com/. [Accessed: 15 Mar 2016].

Appendices

Appendix 1: Evaluation Sheets

Names have been removed for anonymity.



Keyboard Controls

uses Up, Down, a,w,s and d in the keyboard to control the Game. Please mark your feedback n the following questions.

1. How convenient is Keyboard controller to use?

- Extremely convenient
- Very convenient
- Moderately convenient
- Slightly convenient
- Not at all convenient

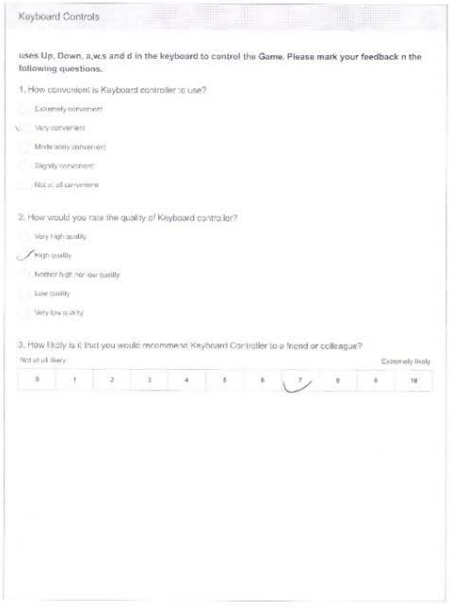
2. How would you rate the quality of Keyboard controller?

- Very high quality
- High quality
- Neither high nor low quality
- Low quality
- Very low quality

3. How likely is it that you would recommend Keyboard Controller to a friend or colleague?

Not at all likely — Extremely likely

0 1 2 3 4 5 6 7 8 9 10



Keyboard Controls

uses Up, Down, a,w,s and d in the keyboard to control the Game. Please mark your feedback n the following questions.

1. How convenient is Keyboard controller to use?

- Extremely convenient
- Very convenient
- Moderately convenient
- Slightly convenient
- Not at all convenient

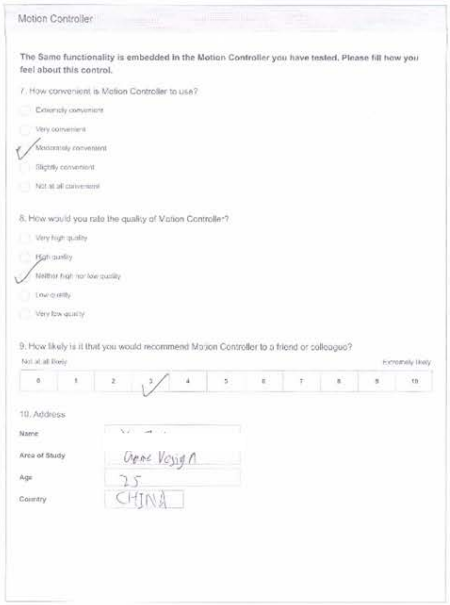
2. How would you rate the quality of Keyboard controller?

- Very high quality
- High quality
- Neither high nor low quality
- Low quality
- Very low quality

3. How likely is it that you would recommend Keyboard Controller to a friend or colleague?

Not at all likely — Extremely likely

0 1 2 3 4 5 6 7 8 9 10



Motion Controller

The Same functionality is embedded in the Motion Controller you have tested. Please fill how you feel about this control.

7. How convenient is Motion Controller to use?

- Extremely convenient
- Very convenient
- Moderately convenient
- Slightly convenient
- Not at all convenient

8. How would you rate the quality of Motion Controller?

- Very high quality
- High quality
- Neither high nor low quality
- Low quality
- Very low quality

9. How likely is it that you would recommend Motion Controller to a friend or colleague?

Not at all likely — Extremely likely

0 1 2 3 4 5 6 7 8 9 10

10. Address

Name: [redacted]

Area of Study: Game Design

Age: 25

Country: CHINA

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:
pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	5	3
production cost	5	1
value for the cost	2	4
Utilization of time in production	3	3
Accuracy	4	4
Realism	4	3
Utilization of creative skills	3	3
Utilization of efforts	4	4
Maintainance	5	2
recomendition	3	3

Please fill your details:
Name:
Area of Studies: Game Technology
University: Bedfordshire University
Date and Signature: 26 Feb 2016
Your comments on commercial motcap system in market:
Need to reduce the price for small company

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:
pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	4	4
production cost	5	4
value for the cost	2	3
Utilization of time in production	4	2
Accuracy	5	1
Realism	4	0
Utilization of creative skills	3	5
Utilization of efforts	2	6 5
Maintainance	1	1
recomendition	0	0

Please fill your details:
Name:
Area of Studies: Compute over development
University: bedfordshire
Date and Signature: 25/2/15
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:
pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	4	3
production cost	1	3
value for the cost	3	4
Utilization of time in production	4	3
Accuracy	4	3
Realism	4	3
Utilization of creative skills	2	4
Utilization of efforts	2	4
Maintainance	4	2
recomendition	4	4

Please fill your details:
Name:
Area of Studies: Animation Technology
University: University of Bedfordshire
Date and Signature: 26/02/16
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:
pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	5	4
production cost	3	5
value for the cost	4	4
Utilization of time in production	5	3
Accuracy	5	5
Realism	3	4
Utilization of creative skills	5	5
Utilization of efforts	5	5
Maintainance	3	4
recomendition	4	3

Please fill your details:
Name:
Area of Studies: BSc Games Development
University: UOB
Date and Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:
pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	4	4
production cost	1	5
value for the cost	1	4
Utilization of time in production	4	3
Accuracy	5	4
Realism	5	3
Utilization of creative skills	2	5
Utilization of efforts	4	5
Maintainance	3	3
recomendition*		

Please fill your details:
Name:
Area of Studies: Computer Animation
University: Bedfordshire (Luton)
Date and Signature: 26/02/16
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:
pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animatinn is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	3	4
production cost	3	3
value for the cost	4	5
Utilization of time in production	5	2
Accuracy	3	4
Realism	3	4
Utilization of creative skills	3	5
Utilization of efforts	4	4
Maintainance	4	3
recomendition	4	3

Please fill your details:
Name:
Area of Studies: game design
University: Bats
Date and Signature: 25/2/16
Your comments on commercial motcap system in market:
No Potential (comments)

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:
pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	3	5
production cost	5	3
value for the cost	4	5
Utilization of time in production	5	5
Accuracy	4	5
Realism	5	5
Utilization of creative skills	2	5
Utilization of efforts	5	5
Maintainance	5	2
recomendition	4	5

Please fill your details:
Name:
Area of Studies: Computer animation
University: UOB
Date and Signature: 26/02/16
Your comments on commercial motcap system in market: N/A.

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:
pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	4	5
production cost	5	4
value for the cost	4	5
Utilization of time in production	5	3
Accuracy	5	4
Realism	5	3
Utilization of creative skills	3	5
Utilization of efforts	5	4
Maintainance	3	5
recomendition	5	3

Please fill your details:
Name:
Area of Studies: [redacted] computer animation
University: [redacted] University of Bedfordshire
Date and Signature: 25.02.2016
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:

pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5 (5 is the highest),

	Pipeline 1	pipeline 2
Animation quality	3	1
production cost	5	3
value for the cost	4	4
Utilization of time in production	4	2
Accuracy	4	3
Realism	4	2
Utilization of creative skills	3	5
Utilization of efforts	5	5
Maintainance	4	3
recomendition	5	4

Please fill your details
Name:
Area of Studies: Electronic engineer.
University: Bed fordshire
Date and Signature: 26.2.16
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:

pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	5	5
production cost	5	1
value for the cost	5	5
Utilization of time in production	4	2
Accuracy	4	4
Realism	5	4
Utilization of creative skills	2	5
Utilization of efforts	5	5
Maintainance	5	3
recomendition	4	4

Please fill your details
Name:
Area of Studies: Motor Racing A1
University: Bucks
Date and Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:

pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5 Short Anim

	Pipeline 1	pipeline 2
Animation quality	5	4
production cost	1	4
value for the cost	2	4
Utilization of time in production	5	3
Accuracy	5	3
Realism	5	4
Utilization of creative skills	3	4
Utilization of efforts	4	3
Maintainance	3	3
recomendition	4	4

Please fill your details
Name:
Area of Studies: Animation
University: UCA
Date and Signature: 26/02/16
Your comments on commercial motcap system in market: Pricy!

Evaluation form for comparing animation done with the tradiational method and using the Motion capture:

The following datas will helps to evaluate wether the Motion capture technology Pipeline reduces the work invested in the production, provides the simpler way of doing animation and also gives the quality in animation.

Proposed Pipeline:

pipeline 1:
System caliberation< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click.
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2:
Tradiational Method by posing and keying each and every moves of the characters. It requires more creativity and time based on the animators skill and creativity.

Evaluation : Please mark the columns from the scale of 5

	Pipeline 1	pipeline 2
Animation quality	4	2
production cost	2	4
value for the cost	5	2
Utilization of time in production	4	1
Accuracy	3	3
Realism	5	2
Utilization of creative skills	3	4
Utilization of efforts	4	1
Maintainance	3	4
recomendition	5	1

Please fill your details
Name:
Area of Studies: Animation
University: UoB
Date and Signature: 26/02/2016
Your comments on commercial motcap system in market: iPi MotionBuilder.

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System calibrations capturings cleaning the track filess solving animation to the bones< tweaking the bone animation to model
system calibration : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The calibration is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	5	3
production cost	1	3
value for the cost	3	4
Utilization of time in production	3	5
Accuracy	5	4
Realism	5	4
Utilization of creative skills	1	1
Utilization of efforts	2	3
Maintainance	3	3
recomendition	3	4

Please fill your details:
Name:
Area of Studies: Comp & Games
University: Bedfordshire
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System calibrations capturings cleaning the track filess solving animation to the bones< tweaking the bone animation to model
system calibration : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The calibration is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	5	4
production cost	1	5
value for the cost	4	2
Utilization of time in production	4	2
Accuracy	5	3
Realism	5	2
Utilization of creative skills	2	5
Utilization of efforts	3	2
Maintainance	2	5
Recommendation recomendition	3	4

Please fill your details:
Name:
Area of Studies:
University:
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System calibrations capturings cleaning the track filess solving animation to the bones< tweaking the bone animation to model
system calibration : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The calibration is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	5	5
production cost	4	4
value for the cost	1	3
Utilization of time in production	1	5
Accuracy	5	5
Realism	5	4
Utilization of creative skills	1	1
Utilization of efforts	1	4
Maintainance	1	4
recomendition	2	2

Please fill your details:
Name:
Area of Studies: DMS
University: Bedfordshire
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System calibrations capturings cleaning the track filess solving animation to the bones< tweaking the bone animation to model
system calibration : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The calibration is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	4	4
production cost	3	5
value for the cost	4	4
Utilization of time in production	3	3
Accuracy	4	3
Realism	4	2
Utilization of creative skills	4	5
Utilization of efforts	4	5
Maintainance	4	4
recomendition	4	4

Please fill your details:
Name:
Area of Studies: Computer Games
University: Uni of Bedfords
Signature:
Your comments on commercial motcap system in market:
Good Stuff, Packed with trigger screening.

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System caliberation> capturing> cleaning the track files> solving animation to the bones> tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The caliberation is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	3	3
production cost	3	3
value for the cost	4	3
Utilization of time in production	3	4
Accuracy	4	4
Realism	4	3
Utilization of creative skills	3	3
Utilization of efforts	4	4
Maintainance	3	4
recomendition	4	3

Please fill your details
Name:
Area of Studies: Cblect
University: beds
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System caliberation> capturing> cleaning the track files> solving animation to the bones> tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The caliberation is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	3	4
production cost	4	3
value for the cost	3	3
Utilization of time in production	3	3
Accuracy	3	4
Realism	4	2
Utilization of creative skills	3	3
Utilization of efforts	4	3
Maintainance	5	3
recomendition	4	4

Please fill your details
Name:
Area of Studies: Science
University: University of Bedfordshire
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System caliberation> capturing> cleaning the track files> solving animation to the bones> tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The caliberation is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	3	4
production cost	2	2
value for the cost	3	3
Utilization of time in production	4	5
Accuracy	5	4
Realism	4	2
Utilization of creative skills	3	2
Utilization of efforts	4	3
Maintainance	3	3
recomendition	5	2

Please fill your details
Name:
Area of Studies:
University:
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System caliberation> capturing> cleaning the track files> solving animation to the bones> tweaking the bone animation to model
system caliberation : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.

pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The caliberation is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	4	3
production cost	4	3
value for the cost	3	4
Utilization of time in production	4	5
Accuracy	4	3
Realism	4	2
Utilization of creative skills	4	4
Utilization of efforts	4	5
Maintainance	3	4
recomendition	4	4

Please fill your details
Name:
Area of Studies:
University: University of Bedfordshire
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System calibration< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system calibration : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The calibration is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	4 ✓	4
production cost	5	5
value for the cost	5	3
Utilization of time in production	3	3
Accuracy	3	5
Realism	4	4
Utilization of creative skills	5	5
Utilization of efforts	5	5
Maintainance	4	4
recomendition	4	5

Please fill your details
Name:
Area of Studies:
University:
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System calibration< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system calibration : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The calibration is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	4	3
production cost	5	1
value for the cost	5	5
Utilization of time in production	3	1
Accuracy	1	1
Realism	1	1
Utilization of creative skills	5	5
Utilization of efforts	5	5
Maintainance	2	2
recomendition	5	3

Please fill your details
Name:
Area of Studies:
University:
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

pipeline 1: Optical Method(Cameras and Trackers)
System calibration< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system calibration : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The calibration is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

	Pipeline 1	pipeline 2
Animation quality	3	3
production cost	3	3
value for the cost	4	3
Utilization of time in production	3	4
Accuracy	4	4
Realism	4	3
Utilization of creative skills	3	3
Utilization of efforts	4	4
Maintainance	3	4
recomendition	4	3

Please fill your details
Name:
Area of Studies: Cblock
University: beds
Signature:
Your comments on commercial motcap system in market:

Evaluation form for comparing animation done with the Optical method and using the Non-Optical MEMS Sensors:

The following data will help to evaluate which technology in Motion capture technology Pipeline reduces the work invested in the production, as well as providing a simpler way of doing animation and the quality in animation. If you are setting a project with your friends, which method do you propose?

Proposed Pipeline:

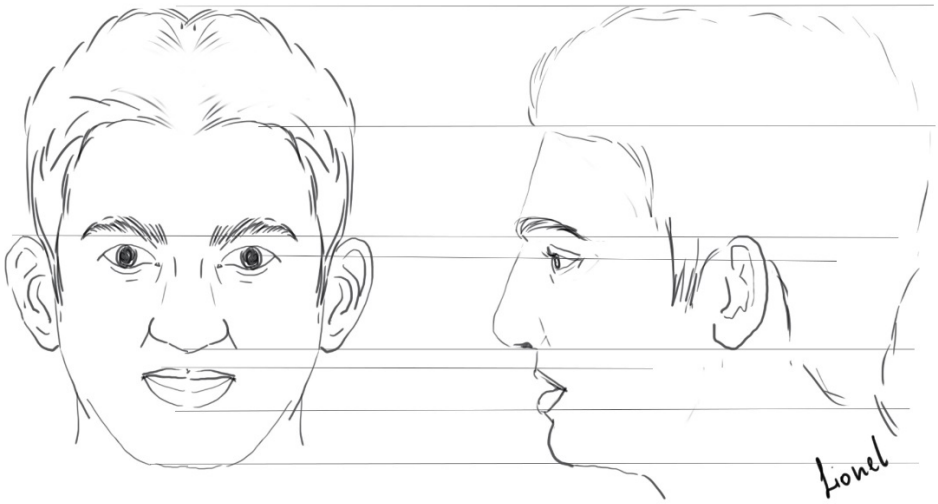
pipeline 1: Optical Method(Cameras and Trackers)
System calibration< capturing< cleaning the track files< solving animation to the bones< tweaking the bone animation to model
system calibration : 5 mins
capturing: 1 mins (maximum)
cleaning track files : 5 to 10 mins per 1 minute animation
solving the bones: requires 20 to 30 mins per character and done once. The next time the animation is solved in one click
Tweaking the animation : 5 mins
for the 1st 1 min animation it takes 1 hrs approximately to set the system. Then it takes less than 15 mins per 1 min animation.
pipeline 2: Non-Optical Method(Sensors and Filters)
Animation can be recorded without the expensive IR Cameras. The calibration is done instantly and the tracks are clean while generated. This method is totally a studio independent.

Evaluation : Please mark the columns from the scale of 1 to 5 (1=Worst and 5=Best)

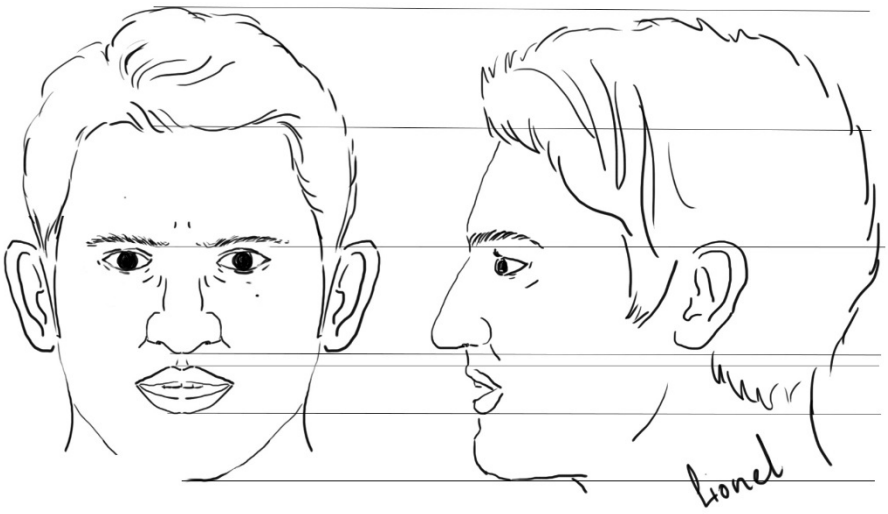
	Pipeline 1	pipeline 2
Animation quality	3	3
production cost	3	3
value for the cost	3	4
Utilization of time in production	4	4
Accuracy	4	4
Realism	4	4
Utilization of creative skills	3	3
Utilization of efforts	4	4
Maintainance	3	3
recomendition	4	4

Please fill your details
Name:
Area of Studies: C block
University: Bedford Shire (utan)
Signature:
Your comments on commercial motcap system in market:

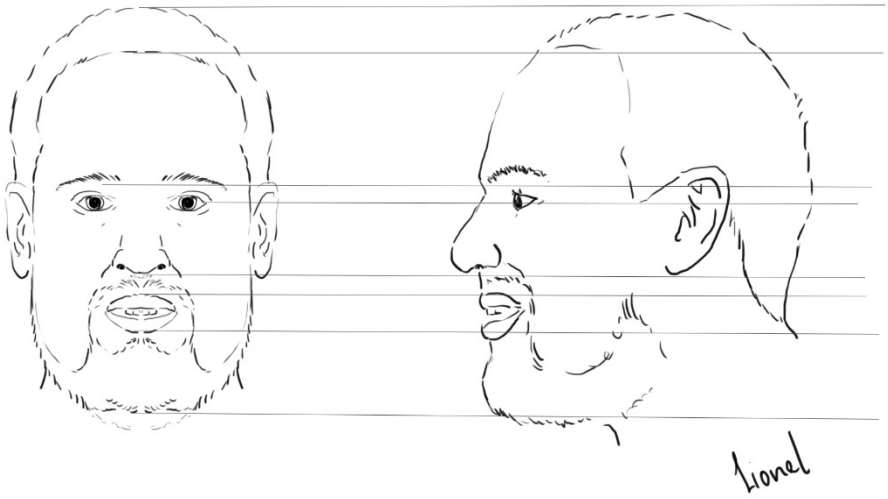
Appendix 2: Character Designs



Captain: KKR

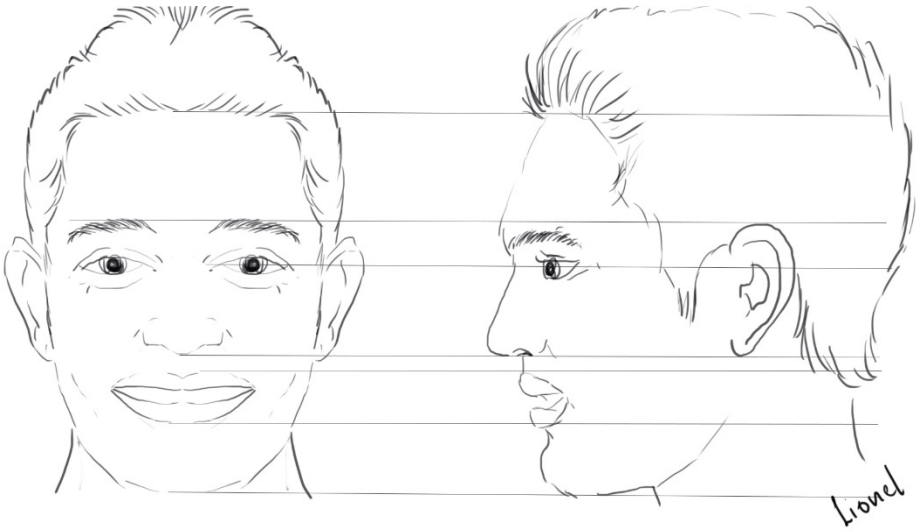


Captain: CSK



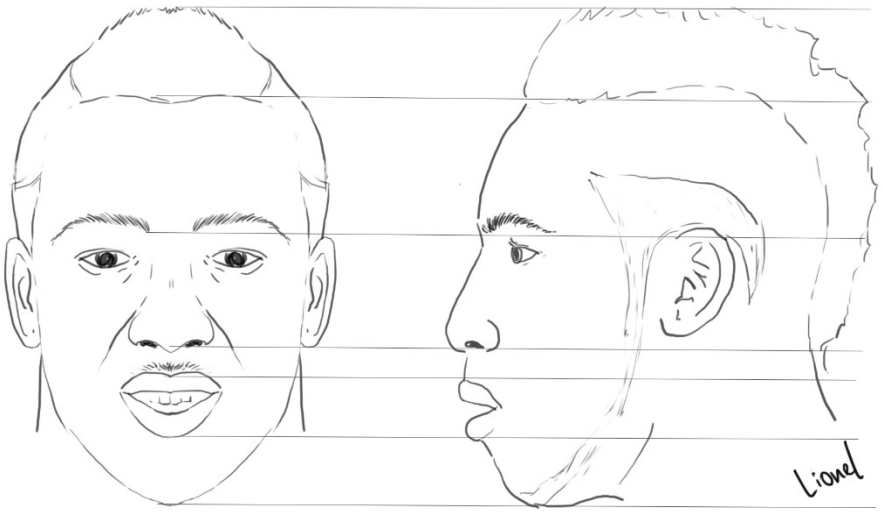
Batsman: KKR

Batsman: CSK



Batsman: KKR

Batsman: CSK



All-rounder: KKR

Bowler: CSK


[1]

Appendix 3: Code Samples

The Unity–Arduino Test Game

Arduino Codes for the potentiometer and buttons [33]

```
const int buttonPin01 = 7;  //specify port
const int buttonPin02 = 8;
void setup()
{
 Serial.begin(9600); //specify baud rate
 pinMode(buttonPin01, INPUT);
 pinMode(buttonPin02, INPUT);
 digitalWrite(buttonPin01, HIGH);
 digitalWrite(buttonPin02, HIGH);
}
void loop()
{
Serial.print(map (analogRead(0),0,1023,0,359)); //Maps input for potentiometer
 Serial.print(",");  // Print the serial monitor
if(digitalRead(buttonPin01) == LOW) // Coding for button press
 {
   Serial.println("LEFT");
 }
 else if(digitalRead(buttonPin02) == LOW)
 {
   Serial.println("RIGHT");
 }
 else
 {
   Serial.println("IDLE");
 }
Serial.flush();
 delay(30);
 }
```

Unity Code for Keyboard controls [34]:

```
public class GooberScript: MonoBehaviour {
// Use this for initialization
            void Start () {
            }
              // Update is called once per frame
            void Update () {
                        if (Input.GetKey ("up")) {        // method to move a game object
                                    GetComponent<Animation> ().Play ("walk");
                                    transform.Translate (0, 0, 0.6f);
                        } else if
                                    (Input.GetKey ("down")) {
                                    GetComponent<Animation> ().Play ("walk");
                                    transform.Translate (0, 0, -0.2f);
                        } else if
                        (Input.GetKey ("right")) {       // method to rotate a game object
                                    GetComponent<Animation> ().Play ("walk");
                                    transform.Rotate (Vector3.up, 5);
                        } else if
                        (Input.GetKey ("left")) {
                                    GetComponent<Animation> ().Play ("walk");
                                    transform.Rotate (Vector3.up, -5);
                        }else

{
                                                GetComponent<Animation> ().Play ("idle");
                                                }
            }
}
```

The Unity–Arduino Gyro Test Game

Tinker kit Gyroscope code for Arduino [33]

The library was simplified such that it suits requirements: the approach needed for implementing the mapping of the raw input and deletion of some unnecessary lines from the library.

Arduino code [33]

```
#include <TinkerKit.h>
TKGyro gyro(I0, I1, TK_X4);  // creating the object 'gyro' that belongs to the 'TKGyro' class
                      // and giving the values to the desired input pins
                     // using the 4x amplified module, insert the TK_4X costant.
 int thirdAngle = 180;   // Made Dummy constant
void setup()
{
  // initialize serial communications at 9600 bps
  Serial.begin(9600);
}
void loop()
{
  Serial.print(map(analogRead(0),0,1023,0,359)); // mapped the Analogue in
  Serial.print(",");
  Serial.print(thirdAngle);
  Serial.print(",");
  Serial.println(map(analogRead(1),0,1023,0,359)); // mapped the Analogue in
  delay(100);
}
```

Unity code for Tinker kit Gyroscope [22]

```
using UnityEngine;
using System.Collections;
using System.IO.Ports;
public class serialRotation: MonoBehaviour {
	SerialPort stream = new SerialPort("COM5", 9600); //Set the port (com5) and the baud rate (9600, is standard
on most devices)
	float[] lastRot = {0,0,0}; //Need the last rotation to tell how far to spin the camera
	Vector3 rot;
	Vector3 offset;
	Vector3 calibration;
	float temp0;
	float temp1;
	float temp2;
	void Start () {
		stream.Open(); //Open the Serial Stream.
		calibration = new Vector3 (180, 180, 180);
	}
	// Update is called once per frame
	void Update () {
		string value = stream.ReadLine(); //Read the information
		string[] vec3 = value.Split(','); //My arduino script returns a 3 part value (IE: 12,30,18)
		if(vec3[0] != "" && vec3[1] != "" && vec3[2] != "") //Check if all values are recieved
		{
```

```
                                        rot = new Vector3(float.Parse(vec3[0]),float.Parse(vec3[1]),float.Parse(vec3[2]));
                                        //Read the information and put it in a vector3
                                        transform.rotation = Quaternion.Slerp(transform.rotation,
                                                                Quaternion.Euler(-(rot.x - temp0),rot.y -temp1,rot.z - temp2),
                                                                Time.deltaTime*3);
//Take the vector3 and apply it to the object this script is applied.
                                        stream.BaseStream.Flush();
//Clear the serial information so we assure we get new information.
                        }
                        if (Input.GetKey ("k")) {
                                        CalibrateSensor();
                        }
                        }
                void CalibrateSensor (){
                        string value = stream.ReadLine();
                        string[] vec3 = value.Split(',');
//My arduino script returns a 3 part  value (IE: 12,30,18)
                        temp0 = float.Parse(vec3[0]) - 180;
                        temp1 = float.Parse(vec3[1]) - 180;
                        temp2 = float.Parse(vec3[2]) - 180;
                }
                void OnGUI()
                {
                        string newString = "Connected: " + transform.eulerAngles;
                        GUI.Label(new Rect(10,10,300,100), newString); //Display new values
                        GUI.Label(new Rect(10,30,300,100), "\t" + rot);
                }
}
//The game has mines that repel the ball back
```

Unity code for mines in the Game [34]

```
using UnityEngine;
Using System.Collections;
public class force : MonoBehaviour {
                public float forceApplied = 100;
                void OnCollisionEnter(Collision collision) {
                        if (collision.gameObject.name == "pSphere1")
                        {
                                Debug.Log("happen");
collision.gameObject.GetComponent<Rigidbody>().AddForce  (forceApplied, forceApplied, forceApplied);
                        }   }   }
```

The Unity and Analogue Joystick

From [22]

```
int xPin = A1;
int yPin = A0;
int buttonPin = 2;
int xPosition = 0;
int yPosition = 0;
int buttonState = 0;
void setup() {
 // initialize serial communications at 9600 bps:
 Serial.begin(9600);
  pinMode(xPin, INPUT);
 pinMode(yPin, INPUT);

 //activate pull-up resistor on the push-button pin
 pinMode(buttonPin, INPUT_PULLUP);
```

```
  // For versions prior to Arduino 1.0.1
  // pinMode(buttonPin, INPUT);
  // digitalWrite(buttonPin, HIGH);
  }
void loop() {
  xPosition = analogRead(xPin);
  yPosition = analogRead(yPin);
  buttonState = digitalRead(buttonPin);
  Serial.print(",");
  Serial.print(xPosition);
  Serial.print(",");
  Serial.print(yPosition);
  Serial.print(",");
  Serial.println(buttonState);
delay(25)
}
```

Unity Coding [22]

The Unity coding is done by reading the string from the array and implementing a switch case statement accordingly. it was tested by applying the translation reacting to the printed input. The statements are:

```
switch(vec3[4])
{
case "0":
transform.position = Vector3.Lerp(transform.position, transform.position + transform.right *  0.2f, 0.2f);
//transform.Translate (0,0.2f,0);
break;
case "1023":
transform.position = Vector3.Lerp(transform.position, transform.position - transform.right *  0.2f, 0.2f);
//transform.Translate (0,-0.2f,0);
break;
}
switch(vec3[5])
{
case "0":
transform.position = Vector3.Lerp(transform.position, transform.position -transform.forward *  0.2f, 0.2f);
//transform.Translate (-0.2f,0,0 );
break;
case "1023":
transform.position = Vector3.Lerp(transform.position, transform.position + transform.forward *  0.2f, 0.2f);
//transform.Translate (0.2f,0,0 );
break;
}
```

Testing I^2C Magnetometer

[33] Copyright (c) 2011 Jeff Rowberg

```
// Arduino Wire library is required if I2Cdev I2CDEV_ARDUINO_WIRE implementation
// is used in I2Cdev.h
#include "Wire.h"

// I2Cdev and HMC5883L must be installed as libraries, or else the .cpp/.h files
// for both classes must be in the include path of your project
#include "I2Cdev.h"
#include "HMC5883L.h"

// class default I2C address is 0x1E
// specific I2C addresses may be passed as a parameter here
```

```
// this device only supports one I2C address (0x1E)
HMC5883L mag;

int16_t mx, my, mz;

#define LED_PIN 13
bool blinkState = false;

void setup() {
   // join I2C bus (I2Cdev library doesn't do this automatically)
   Wire.begin();

   // initialize serial communication
   // (38400 chosen because it works as well at 8MHz as it does at 16MHz, but
   // it's up to you depending on your project)
   Serial.begin(9600);

   // initialize device
   Serial.println("Initializing I2C devices...");
   mag.initialize();

   // verify connection
   Serial.println("Testing device connections...");
   Serial.println(mag.testConnection() ? "HMC5883L connection successful" : "HMC5883L connection failed");

   // configure Arduino LED for
   pinMode(LED_PIN, OUTPUT);
}

void loop() {
   // read raw heading measurements from device
   mag.getHeading(&mx, &my, &mz);

   // display tab-separated gyro x/y/z values
// display tab-separated gyro x/y/z values
   Serial.print("x: ");
   Serial.print(mx);
   Serial.print("  y: ");
   Serial.print(my);
   Serial.print("  Z: ");
   Serial.print(mz);
// To calculate heading in degrees. 0 degree indicates North
   float heading = atan2(my, mx);
   if(heading < 0)
     heading += 2 * M_PI;
   Serial.print(" heading: ");
   Serial.println(heading * 180/M_PI);
   // blink LED to indicate activity
   blinkState = !blinkState;
   digitalWrite(LED_PIN, blinkState);
}
```

Testing I²C MPU 6050

From [33]

```
#define OUTPUT_READABLE_YAWPITCHROLL and
 #define OUTPUT_READABLE_WORLDACCEL
I2Cdev device library code was placed under the MIT license
Copyright (c) 2012 Jeff Rowberg
#include "I2Cdev.h"
#include "MPU6050_6Axis_MotionApps20.h"
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
#include "Wire.h"
```

```
#endif
MPU6050 mpu;
#define OUTPUT_READABLE_YAWPITCHROLL
#define OUTPUT_READABLE_WORLDACCEL
#define address 0x1E
byte mode = 0x02; //mode select register
byte reg = 0x00; //continuous read mode
byte xreg = 0x03; //first data register (1 of 6, MSB and LSB for x, y and z
#define LED_PIN 13
bool blinkState = false;
bool dmpReady = false;
uint8_t mpuIntStatus;
uint8_t devStatus;
uint16_t packetSize;
uint16_t fifoCount;
uint8_t fifoBuffer[64];
Quaternion q;
VectorInt16 aa;
VectorInt16 aaReal;
VectorInt16 aaOld;
VectorInt16 aaWorld;
VectorFloat gravity;
float euler[3];
float ypr[3];

uint8_t teapotPacket[14] = { '$', 0x02, 0,0, 0,0, 0,0, 0,0, 0x00, 0x00, '\r', '\n' };
volatile bool mpuInterrupt = false;     // indicates whether MPU interrupt pin has gone high
void dmpDataReady() {
   mpuInterrupt = true;
}
// Initial Setup
void setup() {
   // join I2C bus (I2Cdev library doesn't do this automatically)
   #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
      Wire.begin();
      TWBR = 24; // 400kHz I2C clock (200kHz if CPU is 8MHz)
   #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
      Fastwire::setup(400, true);
   #endif
// initialize serial communication
   Serial.begin(9600);
   while (!Serial); // wait for Arduino enumeration, others continue immediately
   // initialize device
  // Serial.println(F("Initializing I2C devices..."));
   mpu.initialize();
 // verify connection
   //Serial.println(F("Testing device connections..."));
  // Serial.println(mpu.testConnection() ? F("MPU6050 connection successful") : F("MPU6050 connection failed"));
 // wait for ready
  // Serial.println(F("\nSend any character to begin DMP programming and demo: "));
   //while (Serial.available() && Serial.read()); // empty buffer
   //while (!Serial.available());                 // wait for data
  // while (Serial.available() && Serial.read()); // empty buffer again
 // load and configure the DMP
   //Serial.println(F("Initializing DMP..."));
   devStatus = mpu.dmpInitialize();
 // supply your own gyro offsets here, scaled for min sensitivity
   mpu.setXGyroOffset(220);
   mpu.setYGyroOffset(76);
   mpu.setZGyroOffset(-85);
   mpu.setZAccelOffset(1788); // 1688 factory default for my test chip
//Put the HMC5883 IC into the correct operating mode
Wire.beginTransmission(address); //open communication with HMC5883
Wire.write(mode); //select mode register
Wire.write(reg); //continuous measurement mode
Wire.endTransmission();
 // make sure it worked (returns 0 if so)
   if (devStatus == 0) {
```

```
        // turn on the DMP, now that it's ready
        //Serial.println(F("Enabling DMP..."));
        mpu.setDMPEnabled(true);
        // enable Arduino interrupt detection
       // Serial.println(F("Enabling interrupt detection (Arduino external interrupt 0)..."));
        attachInterrupt(0, dmpDataReady, RISING);
        mpuIntStatus = mpu.getIntStatus();
        // set our DMP Ready flag so the main loop() function knows it's okay to use it
       // Serial.println(F("DMP ready! Waiting for first interrupt..."));
        dmpReady = true;
     // get expected DMP packet size for later comparison
        packetSize = mpu.dmpGetFIFOPacketSize();
    } else {
        // ERROR!
        // 1 = initial memory load failed
        // 2 = DMP configuration updates failed
        // (if it's going to break, usually the code will be 1)
       // Serial.print(F("DMP Initialization failed (code "));
       // Serial.print(devStatus);
      //  Serial.println(F(")"));
    }
 // configure LED for output
    pinMode(LED_PIN, OUTPUT);
}
 // Main Program
void loop() {
  int x,y,z; //triple axis data
    // if programming failed, don't try to do anything
    if (!dmpReady) return;
 // wait for MPU interrupt or extra packet(s) available
    while (!mpuInterrupt && fifoCount < packetSize) {
        // other program behavior stuff here
        // .
        // .
        // .
        // if you are really paranoid you can frequently test in between other
        // stuff to see if mpuInterrupt is true, and if so, "break;" from the
        // while() loop to immediately process the MPU data
        // .
        // .
        // .
    }
// reset interrupt flag and get INT_STATUS byte
    mpuInterrupt = false;
    mpuIntStatus = mpu.getIntStatus();
 // get current FIFO count
    fifoCount = mpu.getFIFOCount();
// check for overflow (this should never happen unless our code is too inefficient)
    if ((mpuIntStatus & 0x10) || fifoCount == 1024) {
        // reset so we can continue cleanly
        mpu.resetFIFO();
        //Serial.println(F("FIFO overflow!"));
// otherwise, check for DMP data ready interrupt (this should happen frequently)
    } else if (mpuIntStatus & 0x02) {
        // wait for correct available data length, should be a VERY short wait
        while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();
 // read a packet from FIFO
        mpu.getFIFOBytes(fifoBuffer, packetSize);
        // track FIFO count here in case there is > 1 packet available
        // (this lets us immediately read more without waiting for an interrupt)
        fifoCount -= packetSize;
  #ifdef OUTPUT_READABLE_YAWPITCHROLL
            // display Euler angles in degrees
            mpu.dmpGetQuaternion(&q, fifoBuffer);
            mpu.dmpGetGravity(&gravity, &q);
            mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);
            //Serial.print("ypr\t");
            Serial.print(ypr[0] * 180/M_PI);
```

```
        Serial.print("\t");
        Serial.print(ypr[1] * 180/M_PI);
        Serial.print("\t");
        Serial.print(ypr[2] * 180/M_PI);
      #endif

      #ifdef OUTPUT_READABLE_WORLDACCEL
        // display initial world-frame acceleration, adjusted to remove gravity
        // and rotated based on known orientation from quaternion
        mpu.dmpGetQuaternion(&q, fifoBuffer);
        mpu.dmpGetAccel(&aa, fifoBuffer);
        mpu.dmpGetGravity(&gravity, &q);
        mpu.dmpGetLinearAccel(&aaReal, &aa, &gravity);
        aaOld = aaWorld;
        mpu.dmpGetLinearAccelInWorld(&aaWorld, &aaReal, &q);
        //Serial.print("aworld\t");
        Serial.print("\t");
        Serial.print((aaWorld.x- aaOld.x)/10);
        Serial.print("\t");
        Serial.print((aaWorld.y - aaOld.y)/10);
        Serial.print("\t");
        Serial.print((aaWorld.z- aaOld.z)/10);
      #endif
//Tell the HMC5883 where to begin reading data
Wire.beginTransmission(address);
Wire.write(xreg); //select register 3, X MSB register
Wire.endTransmission();
// blink LED to indicate activity
      blinkState = !blinkState;
      digitalWrite(LED_PIN, blinkState);
   }
}
```

Madgwick's Algorithm

From [11]

```
#include "MadgwickAHRS.h"
#include <math.h>
// AHRS algorithm update
void Madgwick::update(float gx, float gy, float gz, float ax, float ay, float az, float mx, float my, float mz) {
        float recipNorm;
        float s0, s1, s2, s3;
        float qDot1, qDot2, qDot3, qDot4;
        float hx, hy;
        float _2q0mx, _2q0my, _2q0mz, _2q1mx, _2bx, _2bz, _4bx, _4bz, _2q0, _2q1, _2q2, _2q3, _2q0q2, _2q2q3,
        q0q0, q0q1, q0q2, q0q3, q1q1, q1q2, q1q3, q2q2, q2q3, q3q3;

        // Use IMU algorithm if magnetometer measurement invalid (avoids NaN in magnetometer normalisation)
        if((mx == 0.0f) && (my == 0.0f) && (mz == 0.0f)) {
                updateIMU(gx, gy, gz, ax, ay, az);
                return;
        }

        // Rate of change of quaternion from gyroscope
        qDot1 = 0.5f * (-q1 * gx - q2 * gy - q3 * gz);
        qDot2 = 0.5f * (q0 * gx + q2 * gz - q3 * gy);
        qDot3 = 0.5f * (q0 * gy - q1 * gz + q3 * gx);
        qDot4 = 0.5f * (q0 * gz + q1 * gy - q2 * gx);

        // Compute feedback only if accelerometer measurement valid (avoids NaN in accelerometer normalisation)
        if(!((ax == 0.0f) && (ay == 0.0f) && (az == 0.0f))) {
```

```
// Normalise accelerometer measurement
recipNorm = invSqrt(ax * ax + ay * ay + az * az);
ax *= recipNorm;
ay *= recipNorm;
az *= recipNorm;

// Normalise magnetometer measurement
recipNorm = invSqrt(mx * mx + my * my + mz * mz);
mx *= recipNorm;
my *= recipNorm;
mz *= recipNorm;

// Auxiliary variables to avoid repeated arithmetic
_2q0mx = 2.0f * q0 * mx;
_2q0my = 2.0f * q0 * my;
_2q0mz = 2.0f * q0 * mz;
_2q1mx = 2.0f * q1 * mx;
_2q0 = 2.0f * q0;
_2q1 = 2.0f * q1;
_2q2 = 2.0f * q2;
_2q3 = 2.0f * q3;
_2q0q2 = 2.0f * q0 * q2;
_2q2q3 = 2.0f * q2 * q3;
q0q0 = q0 * q0;
q0q1 = q0 * q1;
q0q2 = q0 * q2;
q0q3 = q0 * q3;
q1q1 = q1 * q1;
q1q2 = q1 * q2;
q1q3 = q1 * q3;
q2q2 = q2 * q2;
q2q3 = q2 * q3;
q3q3 = q3 * q3;

// Reference direction of Earth's magnetic field
hx = mx * q0q0 - _2q0my * q3 + _2q0mz * q2 + mx * q1q1 + _2q1 * my * q2 + _2q1 * mz * q3 - mx *
q2q2 - mx * q3q3;
hy = _2q0mx * q3 + my * q0q0 - _2q0mz * q1 + _2q1mx * q2 - my * q1q1 + my * q2q2 + _2q2 * mz
* q3 - my * q3q3;
_2bx = sqrt(hx * hx + hy * hy);
_2bz = -_2q0mx * q2 + _2q0my * q1 + mz * q0q0 + _2q1mx * q3 - mz * q1q1 + _2q2 * my * q3 - mz
* q2q2 + mz * q3q3;
_4bx = 2.0f * _2bx;
_4bz = 2.0f * _2bz;

// Gradient decent algorithm corrective step
s0 = -_2q2 * (2.0f * q1q3 - _2q0q2 - ax) + _2q1 * (2.0f * q0q1 + _2q2q3 - ay) - _2bz * q2 * (_2bx *
(0.5f - q2q2 - q3q3) + _2bz * (q1q3 - q0q2) - mx) + (-_2bx * q3 + _2bz * q1) * (_2bx * (q1q2 - q0q3) +
_2bz * (q0q1 + q2q3) - my) + _2bx * q2 * (_2bx * (q0q2 + q1q3) + _2bz * (0.5f - q1q1 - q2q2) - mz);
s1 = _2q3 * (2.0f * q1q3 - _2q0q2 - ax) + _2q0 * (2.0f * q0q1 + _2q2q3 - ay) - 4.0f * q1 * (1 - 2.0f *
q1q1 - 2.0f * q2q2 - az) + _2bz * q3 * (_2bx * (0.5f - q2q2 - q3q3) + _2bz * (q1q3 - q0q2) - mx) +
(_2bx * q2 + _2bz * q0) * (_2bx * (q1q2 - q0q3) + _2bz * (q0q1 + q2q3) - my) + (_2bx * q3 - _4bz *
q1) * (_2bx * (q0q2 + q1q3) + _2bz * (0.5f - q1q1 - q2q2) - mz);
s2 = -_2q0 * (2.0f * q1q3 - _2q0q2 - ax) + _2q3 * (2.0f * q0q1 + _2q2q3 - ay) - 4.0f * q2 * (1 - 2.0f *
q1q1 - 2.0f * q2q2 - az) + (-_4bx * q2 - _2bz * q0) * (_2bx * (0.5f - q2q2 - q3q3) + _2bz * (q1q3 -
q0q2) - mx) + (_2bx * q1 + _2bz * q3) * (_2bx * (q1q2 - q0q3) + _2bz * (q0q1 + q2q3) - my) + (_2bx *
q0 - _4bz * q2) * (_2bx * (q0q2 + q1q3) + _2bz * (0.5f - q1q1 - q2q2) - mz);
s3 = _2q1 * (2.0f * q1q3 - _2q0q2 - ax) + _2q2 * (2.0f * q0q1 + _2q2q3 - ay) + (-_4bx * q3 + _2bz *
q1) * (_2bx * (0.5f - q2q2 - q3q3) + _2bz * (q1q3 - q0q2) - mx) + (-_2bx * q0 + _2bz * q2) * (_2bx *
(q1q2 - q0q3) + _2bz * (q0q1 + q2q3) - my) + _2bx * q1 * (_2bx * (q0q2 + q1q3) + _2bz * (0.5f - q1q1
- q2q2) - mz);
recipNorm = invSqrt(s0 * s0 + s1 * s1 + s2 * s2 + s3 * s3); // normalise step magnitude
s0 *= recipNorm;
s1 *= recipNorm;
s2 *= recipNorm;
s3 *= recipNorm;

// Apply feedback step
```

```
                qDot1 -= beta * s0;
                qDot2 -= beta * s1;
                qDot3 -= beta * s2;
                qDot4 -= beta * s3;
        }

        // Integrate rate of change of quaternion to yield quaternion
        q0 += qDot1 * (1.0f / sampleFreq);
        q1 += qDot2 * (1.0f / sampleFreq);
        q2 += qDot3 * (1.0f / sampleFreq);
        q3 += qDot4 * (1.0f / sampleFreq);

        // Normalise quaternion
        recipNorm = invSqrt(q0 * q0 + q1 * q1 + q2 * q2 + q3 * q3);
        q0 *= recipNorm;
        q1 *= recipNorm;
        q2 *= recipNorm;
        q3 *= recipNorm;
}

//---------------------------------------------------------------------------------------------------
// IMU algorithm update

void Madgwick:updateIMU(float gx, float gy, float gz, float ax, float ay, float az) {
        float recipNorm;
        float s0, s1, s2, s3;
        float qDot1, qDot2, qDot3, qDot4;
        float _2q0, _2q1, _2q2, _2q3, _4q0, _4q1, _4q2 ,_8q1, _8q2, q0q0, q1q1, q2q2, q3q3;

        // Rate of change of quaternion from gyroscope
        qDot1 = 0.5f * (-q1 * gx - q2 * gy - q3 * gz);
        qDot2 = 0.5f * (q0 * gx + q2 * gz - q3 * gy);
        qDot3 = 0.5f * (q0 * gy - q1 * gz + q3 * gx);
        qDot4 = 0.5f * (q0 * gz + q1 * gy - q2 * gx);

        // Compute feedback only if accelerometer measurement valid (avoids NaN in accelerometer normalisation)
        if(!((ax == 0.0f) && (ay == 0.0f) && (az == 0.0f))) {

                // Normalise accelerometer measurement
                recipNorm = invSqrt(ax * ax + ay * ay + az * az);
                ax *= recipNorm;
                ay *= recipNorm;
                az *= recipNorm;

                // Auxiliary variables to avoid repeated arithmetic
                _2q0 = 2.0f * q0;
                _2q1 = 2.0f * q1;
                _2q2 = 2.0f * q2;
                _2q3 = 2.0f * q3;
                _4q0 = 4.0f * q0;
                _4q1 = 4.0f * q1;
                _4q2 = 4.0f * q2;
                _8q1 = 8.0f * q1;
                _8q2 = 8.0f * q2;
                q0q0 = q0 * q0;
                q1q1 = q1 * q1;
                q2q2 = q2 * q2;
                q3q3 = q3 * q3;

                // Gradient decent algorithm corrective step
                s0 = _4q0 * q2q2 + _2q2 * ax + _4q0 * q1q1 - _2q1 * ay;
                s1 = _4q1 * q3q3 - _2q3 * ax + 4.0f * q0q0 * q1 - _2q0 * ay - _4q1 + _8q1 * q1q1 + _8q1 * q2q2 +
                _4q1 * az;
                s2 = 4.0f * q0q0 * q2 + _2q0 * ax + _4q2 * q3q3 - _2q3 * ay - _4q2 + _8q2 * q1q1 + _8q2 * q2q2 +
                _4q2 * az;
                s3 = 4.0f * q1q1 * q3 - _2q1 * ax + 4.0f * q2q2 * q3 - _2q2 * ay;
                recipNorm = invSqrt(s0 * s0 + s1 * s1 + s2 * s2 + s3 * s3); // normalise step magnitude
                s0 *= recipNorm;
```

```
                        s1 *= recipNorm;
                        s2 *= recipNorm;
                        s3 *= recipNorm;

                        // Apply feedback step
                        qDot1 -= beta * s0;
                        qDot2 -= beta * s1;
                        qDot3 -= beta * s2;
                        qDot4 -= beta * s3;
                }

                // Integrate rate of change of quaternion to yield quaternion
                q0 += qDot1 * (1.0f / sampleFreq);
                q1 += qDot2 * (1.0f / sampleFreq);
                q2 += qDot3 * (1.0f / sampleFreq);
                q3 += qDot4 * (1.0f / sampleFreq);

                // Normalise quaternion
                recipNorm = invSqrt(q0 * q0 + q1 * q1 + q2 * q2 + q3 * q3);
                q0 *= recipNorm;
                q1 *= recipNorm;
                q2 *= recipNorm;
                q3 *= recipNorm;
}

//---------------------------------------------------------------------------------------------------
// Fast inverse square-root
// See: http://en.wikipedia.org/wiki/Fast_inverse_square_root

float Madgwick::invSqrt(float x) {
                float halfx = 0.5f * x;
                float y = x;
                long i = *(long*)&y;
                i = 0x5f3759df - (i>>1);
                y = *(float*)&i;
                y = y * (1.5f - (halfx * y * y));
                return y;
}
```